INSTITUTO SUPERIOR DE ENGENHARIA DO PORTO
MESTRADO EM ENGENHARIA INFORMÁTICA




Sociedade de Sistemas Multi-Agente para o Estudo de Sistemas de Energia Elétrica

BRÍGIDA CONSTANÇA CORREIA TEIXEIRA
Outubro de 2019

POLITÉCNICO
DO PORTO

Sociedade de Sistemas Multi-Agente para o Estudo de Sistemas de Energia Elétrica

Brígida Constança Correia Teixeira

Dissertação para obtenção do Grau de Mestre em Engenharia Informática, Área de Especialização em Sistemas de Informação e Conhecimento

Orientador: Doutora Isabel Praça
Co-Orientador: Mestre Gabriel Santos

Júri:
Presidente:
Dr. Jonny Smith, Professor, DEI/ISEP

Vogais:
Dr. Jaimie Smith, Professor, DEI/ISEP
Dr. Jones Smith, Professor, DEI/ISEP
Dr. Jagger Smith, Professor, DEI/ISEP

Porto, 13 de Outubro de 2019

«Agir, eis a inteligência verdadeira. Serei o que quiser. Mas tenho que querer o que for. O êxito está em ter êxito, e não em ter condições de êxito. Condições de palácio tem qualquer terra larga, mas onde estará o palácio se não o fizerem ali?»

Fernando Pessoa

Resumo

Os mercados de eletricidade são ambientes complexos e dinâmicos com características muito particulares. Metas ambiciosas impostas pela União Europeia têm vindo a fomentar o aumento da geração baseada em fontes de energia renovável, de natureza intermitente, o que contribui para grandes mudanças no sistema, incluindo a introdução das Redes Elétricas Inteligentes (REI).

O uso de ferramentas de simulação e o estudo de diferentes mecanismos de mercado e das relações entre os seus intervenientes, é essencial. No entanto, um dos principais desafios neste domínio é o desenvolvimento de ferramentas de apoio à decisão que permitam abordar o problema como um todo.

Este trabalho propõe contribuir para o aumento da interoperabilidade entre sistemas heterogéneos, nomeadamente baseados em agentes, dirigidos ao estudo dos mercados de eletricidade, à operação das REI, e à gestão energética do consumidor final. Para tal, é proposto o desenvolvimento de um sistema que facilite a interação entre entidades de naturezas distintas, provenientes de sistemas diversos, nomeadamente através do uso de ontologias.

Palavras-chave: Apoio à Decisão, Definição de Cenários, Interoperabilidade Semântica, Simulação Multi-agente, Sistemas de Energia Elétrica

Abstract

Electricity markets are complex and dynamic environments with very particular characteristics. Ambitious targets imposed by the European Union have been encouraging the growth of generation based on renewable energy sources, with intermittent nature; which contributes to major changes in the system, including the introduction of Smart Grids.

The use of simulation tools, and the study of different market mechanisms and the relationships among their stakeholders, is essential. However, one of the key challenges in this area is the development of decision support tools to address the problem as a whole.

This work proposes to contribute to the increase of the interoperability between heterogeneous systems, namely agent based, to study the electricity markets, the operation of the smart grids, and the energy management of the final consumer. For this, it is proposed the development of a system that facilitates the interaction between entities of different natures, comming from diverse systems, namely through the use of ontologies.

Agradecimentos

Em primeiro lugar, quero agradecer aos meus supervisores Doutora Isabel Praça, Mestre Gabriel Santos e Doutor Tiago Pinto, pelo apoio e dedicação ao longo do meu percurso. São exemplos de sabedoria, excelência, profissionalismo e dedicação, que me motivam a querer fazer mais e melhor.

Quero também agradecer aos meus colegas e amigos do GECAD, pelos momentos de companheirismo, boa disposição, e pelos pequenos momentos que tornam o GECAD uma segunda casa. Um agradecimento especial à Doutora Zita Vale, por todas as oportunidades que me proporcionou.

Agradeço aos meus amigos mais próximos que, apesar da minha ausência e indisponibilidade, fizeram sempre questão de demonstrar o quanto acreditam em mim e o seu afeto.

Agradeço à minha família mais próxima, e em especial quero demonstrar a minha profunda gratidão aos meus pais. Não há limites para agradecer todo o amor, compreensão, e orgulho que transmitem, e por terem permitido com que eu trilhasse o meu próprio caminho e assumisse as minhas decisões.

E por último, agradeço ao meu namorado, Francisco, por ser o melhor companheiro que eu poderia ter, por partilhar comigo todos os momentos, pelo apoio incondicional, e por me inspirar a ser uma pessoa melhor.

Homero escreveu *"Suportável é o trabalho quando muitos compartilham do cansaço"*. Aos que já mencionei e a todos os outros que foram cruzando o meu caminho nesta jornada, o meu mais sincero obrigada.

Conteúdo

Lista de Figuras **xiii**

Lista de Tabelas **xv**

Lista de Algorítmos **xv**

Lista de Código **xviii**

Lista de Símbolos **xix**

1 Introdução **1**
1.1 Motivação . . . 1
1.2 Objetivos . . . 2
1.3 Contribuições . . . 4
1.4 Estrutura da Dissertação . . . 5

2 Contexto e Estado da Arte **7**
2.1 Introdução . . . 7
2.2 Análise de Valor de Negócio . . . 7
2.2.1 Identificação e Análise da Oportunidade . . . 8
2.2.2 Geração e Enriquecimento de Ideias . . . 8
2.2.3 Seleção de Ideias e Definição de Conceito . . . 9
2.2.4 Valor para o Cliente . . . 9
2.2.5 Modelo de Negócio . . . 10
2.2.6 AHP . . . 12
2.3 Agentes e Sistemas Multi-Agente . . . 15
2.3.1 Agente de *Software* . . . 15
2.3.2 Sistemas Multi-Agente . . . 16
2.3.3 Padrões . . . 17
2.4 Comunicação e Interoperabilidade entre Agentes . . . 18
2.4.1 Protocolos de Comunicação . . . 19
2.4.2 Linguagens de Comunicação . . . 20
2.4.3 Ontologias . . . 20
2.4.4 Linguagens de Representação de Conhecimento . . . 25
2.5 Plataformas de Desenvolvimento de Sistemas Multi-agente . . . 25
2.6 Simulação em Sistemas Elétricos de Energia . . . 26
2.6.1 Ferramentas de Simulação . . . 27
2.6.2 Co-Simulação e Simulação Integrada . . . 28
2.6.3 Ferramentas de Co-Simulação e Simulação Integrada . . . 31
2.7 Considerações Finais . . . 34

3 Design e Desenvolvimento da Solução **37**

3.1 Introdução . . . 37
3.2 Tools Control Center . . . 38
3.2.1 Requisitos do Sistema . . . 39
3.2.2 Funcionalidades do Sistema . . . 40
Definição de Ferramentas . . . 41
Definição de Dados de Entrada . . . 42
Simulação e Interoperabilidade . . . 43
Análise e Comparação de Resultados . . . 44
3.2.3 Pressupostos . . . 44
3.2.4 Arquitetura . . . 44
3.3 Modelo Multi-Agente . . . 46
3.3.1 Processo de Criação de Agentes . . . 48
3.3.2 Receção e Interpretação de Mensagens . . . 50
3.3.3 Distribuição de Agentes por Máquinas . . . 52
3.3.4 Simulação . . . 53
3.4 Modelo de Dados . . . 55
3.4.1 Repositório TOOCC . . . 55
Processo de Simulação . . . 55
Programas de Demand Response . . . 56
Tarifas . . . 58
3.4.2 Repositório GECAD . . . 59
3.5 Modelo Semântico . . . 59
3.5.1 Ontologia TOOCC . . . 61
3.5.2 Aplicação das Ontologias . . . 66
Fase de Preparação da Simulação . . . 66
Fase de Simulação . . . 68
Fase de Comparação de Resultados . . . 69
3.6 Interface Gráfica do Utilizador . . . 70
3.7 Considerações Finais . . . 75

4 Casos de Estudo **77**
4.1 Introdução . . . 77
4.2 Caso de Estudo 1 - Interoperabilidade Entre Serviços Web . . . 77
4.2.1 Problema . . . 78
4.2.2 Execução . . . 78
4.2.3 Resultados . . . 86
4.3 Caso de Estudo 2 - Interoperabilidade Entre Sistemas Multi-Agente . . . 90
4.3.1 Problema . . . 90
4.3.2 Execução . . . 90
4.3.3 Resultados . . . 94
4.4 Caso de Estudo 3 - Interoperabilidade Entre Sistemas Multi-Agente e Serviços Web . . . 95
4.4.1 Problema . . . 96
4.4.2 Execução . . . 96
4.4.3 Resultados . . . 100
4.5 Caso de Estudo 4 - Comparação de Resultados . . . 102
4.5.1 Problema . . . 102
4.5.2 Execução . . . 102
4.5.3 Resultados . . . 104

4.6 Considerações Finais 105

5 Conclusões **107**
5.1 Síntese e Conclusões 107
5.2 Trabalho Futuro 109

Bibliografia **111**

Lista de Figuras

2.1 Modelo do NCD (Desenvolvimento do Novo Conceito) (A. Koen et al. 2002). 8
2.2 Modelo canvas . . . 11
2.3 Modelo de referência para gestão de agentes *Foundation for Intelligent Physical Agents* (FIPA) (FIPA 2004) . . . 18
2.4 Classificação das ontologias de acordo com o grau de formalidade (adaptado de Hadzic et al. 2009) . . . 22
2.5 Classificação das ontologias de acordo com o nível de generalidade (adaptado de Hadzic et al. 2009) . . . 22
2.6 Classificação das ontologias de acordo com a estrutura de conceptualização (adaptado de Hadzic et al. 2009) . . . 23
2.7 Classificação das ontologias com base na sua expressividade (adaptado de Hadzic et al. 2009) . . . 24
2.8 Co-simulação (adaptado de Mets, Ojea e Develder 2014) . . . 29
2.9 Simulação integrada (adaptado de Mets, Ojea e Develder 2014) . . . 29
2.10 Arquitetura da ferramenta *Electric Power and Communication Synchronizing Simulator* (EPOCHS) (adaptado de Mets, Ojea e Develder 2014) . . . 31
2.11 Arquitetura da ferramenta *Global Event-driven Co-simulation* (GECO) (adaptado de Mets, Ojea e Develder 2014) . . . 32
2.12 Comunicação abstrata entre a *framework* de co-simulação e o Mosaik (adaptado de Schütte 2013) . . . 33

3.1 Processo geral para a simulação de cenários *Tools Control Center* (TOOCC) 39
3.2 Diagrama *Unified Modeling Language* (UML) de Casos de Uso . . . 40
3.3 Perspectiva geral do conceito da ferramenta TOOCC . . . 41
3.4 Exemplo de comunicação entre ferramentas externas e a TOOCC . . . 43
3.5 Arquitetura geral da ferramenta TOOCC . . . 46
3.6 Modelo Multi-Agente . . . 47
3.7 Diagrama UML de Sequência relativo à perspetiva geral das comunicações do sistema TOOCC para a criação dos agentes . . . 49
3.8 Arquitetura base de um agente . . . 50
3.9 Fluxograma do comportamento MessageServerBehaviour implementado pelos agentes . . . 51
3.10 Diagrama UML de Classes (simplificado) relativo à caixa de entrada na receção de mensagens por parte dos agentes . . . 52
3.11 Diagrama representativo da distribuição dos Remote Mobility Agent (RMobA) pelas máquinas . . . 52
3.12 Diagrama de Sequência UML relativo à perspectiva geral das comunicações entre agentes para a simulação . . . 54
3.13 Modelo de dados UML para o processo de simulação . . . 56
3.14 Modelo de dados UML para o modelo de criação de programas de *Demand Response* . . . 57

3.15 Modelo de dados UML para o modelo de Tarifação de Energia 58
3.16 Ontologias de domínio para interoperabilidade 60
3.17 Classes da Ontologia TOOCC 61
3.18 Propriedades dos Dados e dos Objetos da Ontologia TOOCC 61
3.19 Ontologia TOOCC 62
3.20 Comunicações com recurso a ontologias para a construção do Sistema Multi-Agente (SMA) 66
3.21 Comunicações com recurso a ontologias para processo de simulação 68
3.22 Comunicações com recurso a ontologias para o processo de comparação de resultados 69
3.23 Página inicial 71
3.24 Página de criação de simulações 71
3.25 Página de criação de cenários 72
3.26 Página de seleção e configuração de serviços 73
3.27 Página de configuração de comparações 73
3.28 Página de execução de simulações 74
3.29 Página de visualização de resultados 74
3.30 Página de visualização do modelo de DR 75

4.1 Perspetiva geral da execução do primeiro caso de estudo - Service-to-Service (S2S) - pela ferramenta TOOCC 79
4.2 Resultado da otimização dos recursos dos consumidores e produtores para o cenário S2S 87
4.3 Resultado da otimização dos recursos dos consumidores e produtores para o cenário S2S 88
4.4 Resultado da agregação quando são criados 3 *custers* para o cenário S2S . 89
4.5 Resultado do algoritmo de remuneração de um consumidor para o cenário S2S 89
4.6 Perspetiva geral da execução do segundo caso de estudo - MAS-to-MAS (MAS2MAS) - pela ferramenta TOOCC 91
4.7 Resultado da relação entre custo e lucro para o participante vendedor de energia nº56, considerado no cenário MAS2MAS 94
4.8 Resultado da relação entre custo e lucro para o participante vendedor de energia nº58, considerado no cenário MAS2MAS 95
4.9 Perspetiva geral da execução do terceiro caso de estudo - Service-to-MAS (S2MAS) - pela ferramenta TOOCC 96
4.10 Resultado da produção de energia segundo a perspetiva do agregador, considerando o cenário S2MAS 100
4.11 Resultado do escalonamento do consumo de energia da rede, considerando o cenário S2MAS 101
4.12 Resultado da aplicação dos cortes de energia ao nível do Facility Manager Agent (FM), considerando o cenário S2MAS 101
4.13 Perspetiva geral da execução do quarto caso de estudo - comparação de resultados - pela ferramenta TOOCC 102
4.14 Perspetiva geral da execução do quarto caso de estudo - comparação de resultados - pela ferramenta TOOCC 104

Lista de Tabelas

2.1 Proposta longitudinal de valor 10
2.2 Avaliação para o critério Legibilidade. 13
2.3 Avaliação para o critério Complexidade. 13
2.4 Avaliação para o critério Performance. 14
2.5 Avaliação para o critério Utilização. 14
2.6 Prioridade Composta das alternativas. 14
2.7 Camadas genéricas para a interoperabilidade (Nguyen et al. 2017) 30
2.8 Sumário da avaliação das ferramentas de co-simulação 34

3.1 Descrição das propriedades dos objetos da ontologia TOOCC recorrendo a sintaxe Description Logic (DL) 63
3.2 Descrição das propriedades dos dados da ontologia TOOCC recorrendo a sintaxe DL 63
3.3 Descrição das propriedades das classes da ontologia TOOCC recorrendo a sintaxe DL 64

Lista de Código

4.1 Prefácio da Base de Conhecimento do cenário 1 (Turtle) 79
4.2 Base de Conhecimento com a configuração geral do cenário 1 (Turtle) . . 79
4.3 Base de Conhecimento com perspetiva geral da configuração da primeira fase de execução do cenário 1 (Turtle) 80
4.4 Versão adaptada da Base de Conhecimento com o modelo de entrada da primeira fase de execução do cenário 1 (Turtle) 81
4.5 Versão adaptada da Base de Conhecimento com um exemplo dos dados transmitidos no modelo de entrada da primeira fase de execução do cenário 1 (Turtle) 81
4.6 Versão adaptada da Base de Conhecimento com um exemplo da reutilização dos conceitos entre modelos de dados entre as fases de execução do cenário 1 (Turtle) 82
4.7 Base de Conhecimento com perspetiva geral da configuração da segunda fase de execução do cenário 1 (Turtle) 84
4.8 Versão adaptada da Base de Conhecimento com o modelo de entrada da segunda fase de execução do cenário 1 (Turtle) 84
4.9 Exemplo do modelo de saída da segunda fase de execução do cenário 1 (Turtle) 84
4.10 Base de Conhecimento com perspetiva geral da configuração da terceira fase de execução do cenário 1 (Turtle) 85
4.11 Versão adaptada da Base de Conhecimento com o modelo de entrada da terceira fase de execução do cenário 1 (Turtle) 85
4.12 Exemplo do modelo de saída da terceira fase de execução do cenário 1 (Turtle) 86
4.13 Prefácio da Base de Conhecimento do cenário 2 (Turtle) 91
4.14 Base de Conhecimento com a configuração geral do cenário 2 (Turtle) . . 92
4.15 Base de Conhecimento com perspetiva geral da configuração da primeira fase de execução do cenário 2 (Turtle) 92
4.16 Base de Conhecimento com perspetiva geral da configuração da segunda fase de execução do cenário 2 (Turtle) 93
4.17 Base de Conhecimento com perspetiva geral configuração da terceira fase de execução do cenário 2 (Turtle) 94
4.18 Base de Conhecimento com a configuração geral do cenário 3 (Turtle) . . 97
4.19 Base de Conhecimento com perspetiva geral da configuração da primeira fase de execução do cenário 3 (Turtle) 97
4.20 Base de Conhecimento com perspetiva geral da configuração da segunda fase de execução do cenário 3 (Turtle) 98
4.21 Base de Conhecimento com perspetiva geral da configuração da segunda fase de execução do cenário 3 (Turtle) 99
4.22 Base de Conhecimento com perspetiva geral da terceira da primeira fase de execução do cenário 3 (Turtle) 100
4.23 Base de Conhecimento com perspetiva geral da configuração do cenário 4 (Turtle) 102

4.24 Adaptação da Base de Conhecimento da configuração da primeira fase de execução do cenário 4 (Turtle) 103

Lista de Símbolos

$\top$	Conceito especial com cada indivíduo como uma instância
$\bot$	Conceito vazio
$\sqcap$	Intersecção ou conjunção de conceitos
$\exists$	Restrição existencial
$\sqsubseteq$	Conceitos inclusos

Capítulo 1

Introdução

1.1 Motivação

Os Mercados de Energia Elétrica (MEE) são ambientes complexos que estão em constante evolução e adaptação às necessidades da sociedade, procurando criar um ambiente cada vez mais competitivo (Shahidehpour, Yamin e Z. Li 2002). Como resultado, os MEE tornaram-se ambientes mais dinâmicos, onde o elevado número de entidades envolvidas e suas interações originam uma crescente imprevisibilidade. Atualmente, o setor energético enfrenta uma nova mudança de paradigma devido à integração das Fontes de Energia Renováveis (FER), que surgem como resposta à necessidade de redução da emissão de gases com efeito de estufa, promovendo a independência energética, redução de custos e minimização do impacto ambiental (Knieps 2013; Wüstenhagen e Menichetti 2012).

A Comissão Europeia criou as metas 20-20-20 (European Commission 2013, 2017), que estabelecem medidas que contribuem para a mudança da legislação no setor da energia até 2020. Este programa almeja atingir três objetivos: (i) redução de 20% das emissões de CO_2, comparativamente aos níveis verificados em 1990; (ii) aumento da eficiência energética em 20%; e aumentar a utilização das FER de forma a que estas representem 20% da produção de energia na União Europeia. Como resultado, surge a implementação em grande escala da Produção Distribuída de Energia (PDE) (Jain et al. 2017), que consiste na produção descentralizada de energia, e que tem especial impacto ao nível do consumidor. Segundo Georgilakis e Hatziargyriou 2013, a PDE contribui para a aplicação de políticas energéticas mais competitivas, diversificação de recursos energéticos, redução de custos operacionais e aumento da qualidade de serviço para o consumidor final. No entanto, a adaptação dos Sistemas Elétricos de Energia (SEE) à natureza intermitente das FER e a atualização dos mesmos de forma a suportar a penetração da PDE levantam novos desafios que se traduzem em alterações profundas no comportamento das entidades envolvidas.

Neste contexto, é indispensável a utilização de ferramentas de simulação desenvolvidas com o propósito de analisar e estudar os diversos domínios dos SEE, uma vez que permitem às entidades participantes lidar com a imprevisibilidade e complexidade do setor. Os simuladores baseados em tecnologia multi-agente possuem particularidades que fazem com que sejam ferramentas indicadas para o estudo dos SEE, nomeadamente pelo seu caráter distribuído, que permite modelar mais facilmente os diversos sistemas e entidades, bem como as suas restrições, além de tornar o modelo mais próximo da realidade, decompondo o problema em módulos de complexidade mais reduzida (Santos, Pinto, Morais et al. 2015). Embora na literatura existam diversas ferramentas para o estudo dos SEE, grande parte destas apresentam a lacuna de apenas permitirem a resolução de problemas em domínios específicos. Desta forma, não é possível refletir com realismo a tendência da evolução do setor, onde os

componentes autónomos surgem de forma integrada para a resolução de um problema mais abrangente.

Assim, surge a necessidade da criação de soluções que permitam estudos mais completos e abrangentes através da co-simulação de ferramentas heterogéneas dedicadas ao estudo de diferentes áreas dos SEE, conferindo mais fiabilidade e realismo à simulação. Para isso, é necessário criar mecanismos de comunicação para que estas ferramentas sejam capazes de trocar conhecimento, ultrapassando as suas características heterogéneas (ex.: linguagem de programação em que são desenvolvidos, diferentes resoluções temporais, sincronização de eventos e processos, entre outros).

É possível encontrar na literatura alguns trabalhos que já identificaram esta necessidade e apresentam soluções que permitem a co-simulação de ferramentas de simulação, tais como Mosaik (Scherfk 2018), *Global Event-driven Co-simulation* (GECO) (Lin, Veda et al. 2012) e *Electric Power and Communication Synchronizing Simulator* (EPOCHS) (Hopkinson et al. 2006). Todavia, após uma análise das mesmas, é possível compreender que estas não permitem a execução de cenários dinâmicos e com diferentes características de forma automática. Ou seja, para mudar a estrutura do cenário, é necessário proceder à reconfiguração e reprogramação da simulação, tornando o processo muito pouco flexível.

O Grupo de Investigação em Engenharia e Computação Inteligente para a Inovação e o Desenvolvimento (GECAD)[1] é uma unidade de Investigação e Desenvolvimento (IED) sediada no Instituto Superior de Engenharia do Instituto Politécnico do Porto (ISEP-IPP), e uma das suas linhas de investigação prende-se ao estudo de SEE. Neste âmbito, a unidade de investigação vem a desenvolver diversos simuladores multi-agente que se dedicam ao estudo de diferentes áreas dos SEE. Alguns exemplos são: *Multi-Agent Simulator for Competitive Electricity Markets* (MASCEM) (Santos, Pinto, Praça et al. 2016b), *Adaptive Learning Strategic Bidding System* (ALBidS) (Pinto et al. 2014) e *Multi-Agent Smart Grid Simulation Platform* (MASGriP) (Oliveira et al. 2012). Nos últimos anos, a instituição tem como um dos seus objetivos a criação de mecanismos que permitam a comunicação entre estes sistemas e a sua co-simulação, de forma a ser possível efetuar estudos que abrangem vários domínios e a análise de resultados tanto numa perspetiva micro (do ponto de vista individual de cada subsistema), assim como numa perspetiva macro, resultante da interação e cooperação de cada subsistema. Neste sentido, começaram a ser desenvolvidas ontologias[2], que pretendem estabelecer a interoperabilidade entre sistemas heterogéneos (Santos, Pinto, Praça et al. 2016a). Apesar dos simuladores já integrarem as ontologias desenvolvidas, ainda é necessária a criação de uma ferramenta que permita a configuração da co-simulação de forma dinâmica, automática e simples, bem como a análise dos resultados, tirando partido das mesmas.

1.2 Objetivos

Esta dissertação propõe a conceção de uma ferramenta multi-agente capaz de criar, simular e analisar cenários abrangentes aos vários domínios dos SEE, através da interoperabilidade entre ferramentas de simulação heterogéneas desenvolvidas pelo GECAD, recorrendo ao uso de ontologias. Assim, é possível a integração estratégica das capacidades individuais de cada ferramenta, permitindo a resolução de problemas com maior cobertura, dado que cenários

[1] GECAD Website - `http://www.gecad.isep.ipp.pt/`

[2] GECAD IES - `http://www.gecad.isep.ipp.pt/ontologies/ies/`

mais complexos permitem atingir resultados com maior qualidade e fidelidade. Por outro lado, a ferramenta deverá também disponibilizar uma interface gráfica, para que o utilizador seja capaz de preparar e acompanhar todo o processo.

Para permitir a execução de cenários com diferentes características, um dos requisitos essenciais prende-se a um elevado grau de flexibilidade na configuração dos mesmos. Deste modo, é necessário que os cenários possam variar de complexidade, ou seja, que resultem da combinação de diversas ferramentas, ou apenas uma delas. Para além disto, com o intuito de lhes conferir mais realismo, estes devem incluir modelos representativos dos SEE (como por exemplo perfis de consumidores, tarifas de preços de eletricidade, entre outros), bem como a utilização de histórico relativo a dados reais (consumos, produção, preços de mercado, etc), que podem ser personalizados pelo utilizador. No entanto, uma vez que a configuração de várias ferramentas e cenários exige o preenchimento de um elevado número de variáveis, é importante simplificar e automatizar a interação com o utilizador, de forma a que esta não se torne uma tarefa com grande dificuldade.

Na fase de simulação, o sistema deve permitir a distribuição automática dos seus agentes pelas diversas máquinas disponíveis no domínio. Assim, além de suportar a execução de simulações que requerem uma elevada capacidade de processamento, também é possível que o agente encontre a máquina que disponibiliza o software necessário para a execução da sua tarefa. Complementarmente, deve ser permitida a simulação de vários cenários em simultâneo, para poupar tempo ao utilizador e permitir a comparação automática de resultados.

Através da análise dos resultados é pretendido disponibilizar possíveis conclusões dos cenários simulados, sob a forma de gráficos e tabelas, com a finalidade de fornecer apoio à decisão ao utilizador. Estas devem considerar os resultados de cada sistema que compõe o cenário configurado, bem como comparações diretas com outros cenários. Este processo deve ser o mais inteligente possível, mas o utilizador também poderá configurar e personalizar novas comparações e gráficos. De modo a simplificar e automatizar o processo de comparação, este irá tirar partido do uso de ontologias.

Considerando as especificações mencionadas, os objetivos a cumprir são:

1. Estudar as soluções atuais que permitem a interoperabilidade entre sistemas heterogéneos na área dos SEE e apurar as suas limitações;
2. Desenvolver uma ferramenta multi-agente que permita a interoperabilidade entre sistemas com características heterogéneas, através do uso de ontologias;
3. Desenvolvimento de novas ontologias que permitam modelar os conceitos da ferramenta de forma a permitir a interoperabilidade e análise de resultados;
4. Conceção de uma interface gráfica que possibilita o utilizador configurar todas as etapas do processo de forma simples;
5. Experimentar e validar a solução desenvolvida através de casos de estudo que simulam cenários realistas, com recurso a diferentes modelos, dados históricos reais e ferramentas ou serviços.

1.3 Contribuições

O trabalho desenvolvido no âmbito desta dissertação é diretamente suportado por dois projetos, financiados pela União Europeia (UE), no âmbito do programa de desenvolvimento H2020 e pela Fundação para a Ciência e a Tecnologia (FCT):

- *Enabling Demand Response for short and real-time Efficient And Market Based Smart Grid Operation* (DREAM-GO)[3] (*H2020-MSCA-RISE Grant Agreement no. 641794*);
- Multi-Agent Systems SemantiC Interoperability for simulation and dEcision supporT in complex energY systems (MAS-Society)[4]. (*PTDC/EEI-EEE/28954/2017*)

Além destes, durante a execução destes projetos foram publicadas as seguintes contribuições científicas, resultantes deste trabalho:

- Publicação em capítulo de livro:
  - *Teixeira B., Pinto T., Santos G., Praça I., Vale Z. "Tools Control Center to Enable the Joint Simulation of Multi-agent Systems". In: De la Prieta F. et al. (eds) Trends in Cyber-Physical Multi-Agent Systems. The PAAMS Collection - 15th International Conference, PAAMS 2017. PAAMS 2017. Advances in Intelligent Systems and Computing, vol 619. Springer, Cham.*
- Publicação em atas de encontros científicos:
  - *Teixeira B., Silva F., Pinto T., Santos G., Praça I., and Vale Z., "TOOCC: Enabling heterogeneous systems interoperability in the study of energy systems,"2017 IEEE Power & Energy Society General Meeting, Chicago, IL, 2017, pp. 1-5.*
    - *Teixeira B., Silva F., Pinto T., Santos G., Praça I., and Vale Z., "Demonstration of Tools Control Center for Energy Systems Simulation," in 16th International Conference on Practical Applications of Agents and Multi-Agent Systems. 2018.*
  - Artigos em revista:
    - *Teixeira B., Pinto T., Silva F., Santos G., Praça I., and Vale Z., "Multi-Agent Decision Support Tool to Enable Interoperability Between Heterogeneous Systems"Applied Sciences. 2018.*

E ao seguinte prémio de excelência científica:

- *IBM Award of Scientific Excellence, 3rd Prize to the Best Demonstration presented during the 16th International Conference on Practical Applications of Agents and Multi-Agent Systems (PAAMS'18): Demonstration of Tools Control Center for Multi-Agent Energy Systems Simulation, Brigida Teixeira, Francisco Silva, Tiago Pinto, Gabriel Santos, Isabel Praça, and Zita Vale*

[3] DREAM-GO - `http://dream-go.ipp.pt/`
[4] MAS-Society - `http://www.gecad.isep.ipp.pt/MAS-Society/`

1.4 Estrutura da Dissertação

Esta dissertação é composta por cinco capítulos. Após a secção introdutória atual, o Capítulo Capítulo 2 apresenta o panorama geral dos atuais desafios do setor energético, e de que forma esta dissertação pode contribuir para a comunidade científica. Adicionalmente, para uma melhor compreensão dos conceitos abordados com este trabalho, é apresentado o estado da arte das ferramentas multi-agente, da interoperabilidade entre agentes e da sua aplicabilidade aos sistemas de elétricos de energia. Além disso, ainda é feito um estudo de quais as soluções existentes que permitem a interoperabilidade entre ferramentas neste setor.

O Capítulo 3 apresenta o trabalho desenvolvido no âmbito desta dissertação. Este começa por apresentar o conceito da solução proposta, quais são os seus objetivos e funcionalidades, e a arquitetura proposta. De seguida, são apresentados os diferentes módulos que compõem o sistema, designadamente o modelo multi-agente que consiste no motor da solução, o modelo de dados que alimenta o sistema, o modelo semântico como método para permitir a interoperabilidade, e por fim a interface gráfica para estabelecer interação com o utilizador.

O Capítulo 4 apresenta quatro casos de estudo com a finalidade de testar e demonstrar o funcionamento da aplicação. Estes testes consideram diferentes problemas com necessidades distintas, e que utilizam ferramentas com características próprias.

No final do documento, no Capítulo 5 são apresentadas as conclusões que resultam do trabalho desenvolvido e sugestão de novas contribuições para trabalho futuro.

Capítulo 2

Contexto e Estado da Arte

2.1 Introdução

Este capítulo é constituído pelas seguintes secções: análise de valor, que faz uma breve contextualização do problema e surgimento da proposta; agentes e Sistema Multi-Agente (SMA); comunicação e interoperabilidade entre agentes, onde são abordados conceitos importantes para a compreensão do trabalho proposto; plataformas de desenvolvimento de SMA; e simulação em SEE, nomeadamente a apresentação de ferramentas semelhantes à proposta nesta dissertação, e simuladores que podem ser incluídos no âmbito deste trabalho.

2.2 Análise de Valor de Negócio

Num mundo marcado pela globalização, competitividade e mudança, o processo de inovação é uma ferramenta vital para a sobrevivência das empresas (Barkema, Baum e Mannix 2002). Embora a melhoria de produtos, serviços ou processos permita fazer face aos obstáculos impostos pelo mercado, para muitas organizações este processo ainda é de difícil adoção. Assim, torna-se importante desenhar modelos de inovação bem definidos de forma a diminuir a complexidade da sua utilização, bem como o risco e indecisão (Desouza et al. 2009).

O processo de inovação é caracterizado por três áreas, nomeadamente o *Fuzzy Front End* (FFE) (Linha da Frente da Inovação), o *New Product Development* (NPD) (Desenvolvimento de Novos Produtos) e a Comercialização (A. Koen et al. 2002). Contudo, segundo os autores, é na primeira que surgem as maiores oportunidades para a melhoria de processos nas organizações, embora este possa ser um processo difuso. Deste modo, A. Koen et al. 2002 propuseram um novo modelo, cujo objetivo é definir o fluxo e elementos chave do FFE chamado *New Concept Development* (NCD).

O NCD é constituído por três componentes, representadas na Figura 2.1. Elas são:

1. Motor (engine): composto pela liderança, cultura e estratégia da organização.
2. Área interna: identificada pelos cinco elementos controláveis do FFE (identificação da oportunidade; análise da oportunidade; geração e enriquecimento de ideias, seleção de ideias e conceito)
3. Fatores de influência: elementos externos às organizações que não são controláveis por elas. Podem ser definidos por canais de distribuição, fatores políticos, legislativos, económicos, entre outros.

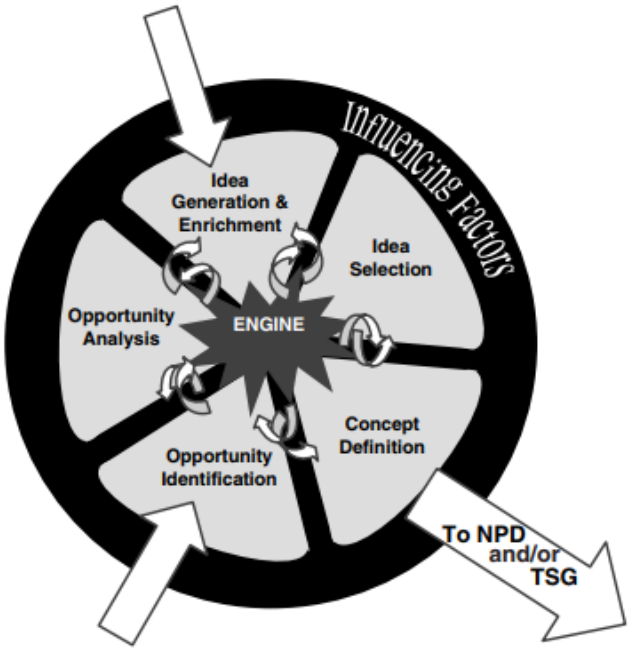

Figura 2.1: Modelo do NCD (Desenvolvimento do Novo Conceito) (A. Koen et al. 2002).

2.2.1 Identificação e Análise da Oportunidade

Os SEE são ambientes complexos que estão em constante adaptação às necessidades da sociedade. Nos últimos anos, normas impostas pela UE no setor energético incentivaram o crescimento significativo da utilização das FER, com o objetivo de reduzir o impacto negativo causado pela emissão de gases de estufa. Como resultado, o setor energético alterou-se por completo com a implementação da microgeração, levando a que este se tornasse ainda mais complexo, dinâmico e intermitente. Neste contexto, surgiram conceitos como Rede Elétrica Inteligente (REI), *Demand Response* (DR) e empoderamento do consumidor.

Assim, surge a necessidade de fornecer ferramentas de apoio à decisão, que ao estudar o comportamento dos vários domínios dos SEE e das suas entidades participantes, permitem lidar com a incerteza e complexidade destes. São várias as entidades que beneficiam deste tipo de ferramentas, nomeadamente os consumidores, os produtores, os operadores de rede, entre outros.

No entanto, a grande maioria das ferramentas existentes são focadas no estudo de domínios específicos, não sendo capazes de efetuar estudos mais completos que traduzem o comportamento do sistema real, onde os sistemas estão interligados e dependentes entre si. Com isto, surge a identificação da oportunidade da criação de uma ferramenta de simulação que permita a simulação de cenários mais complexos, abrangendo vários domínios dos SEE.

2.2.2 Geração e Enriquecimento de Ideias

Para poder aumentar a abrangência do domínio dos cenários de simulação existem duas alternativas: (i) construir um simulador que seja capaz de simular esses domínios; ou (ii) utilizar as ferramentas de simulação já existentes. A primeira opção implica um grande

esforço de desenvolvimento uma vez que se pode revelar uma tarefa demasiado complexa. Por sua vez, a reutilização de ferramentas já existentes pode trazer vários benefícios, tais como o reduzido esforço no desenvolvimento (uma vez que apenas é necessário interligar as ferramentas) e, por outro lado, as ferramentas já estão devidamente testadas.

Para criar um sistema que seja capaz de simular cenários que resultam da interação de vários sistemas, também é necessário efetuar a sua configuração. Esta tarefa pode ter um elevado grau de complexidade, pois depende do número de sistemas envolvidos e propósito do cenário em si, ou seja, é necessário configurar e em preencher um elevado número de variáveis. Por esta razão, é importante que o processo seja simplificado, nomeadamente através da disponibilização de modelos já previamente definidos.

Uma vez que são considerados vários domínios, a análise dos resultados deve considerar tanto uma perspetiva individual do domínio, assim como uma perspetiva mais abrangente, resultante da combinação de diferentes domínios.

2.2.3 Seleção de Ideias e Definição de Conceito

Tendo em conta a fase de geração de ideias do processo NCD, o conceito proposto nesta tese é a elaboração de uma ferramenta de simulação multi-agente, que permite a interoperabilidade entre sistemas heterogéneos, com o objetivo de criar, simular e analisar cenários que incorporam os vários domínios dos SEE.

2.2.4 Valor para o Cliente

O conceito de valor, pode adquirir várias definições consoante o seu contexto. Pode ser considerado como sendo a necessidade, desejo, interesse, crenças, atitudes e preferências (Nicola, E. P. Ferreira e J. J. P. Ferreira 2012). No entanto, é possível considerar que este identifica o benefício que algo oferece, tendo em conta determinados critérios. Na verdade, a indefinição relativa a este conceito reside na definição desses mesmos critérios.

Woodall 2003 define valor para o cliente como sendo a perceção pessoal da vantagem originada da associação do cliente com a oferta que uma organização oferece. Além disso, também pode ser compreendido como a combinação ponderada entre benefícios e sacrifícios.

Por sua vez, o conceito de valor percebido refere-se à perceção do cliente em relação ao custo-benefício de uma determinada oferta (Lindgreen e Wynstra 2005). Este tipo de valor é reconhecido pelas organizações, pois é através dele que o cliente é influenciado a adquirir a oferta.

A proposta de valor deste trabalho reside na criação de uma ferramenta capaz de interoperar sistemas heterogéneos, com vista em efetuar simulações de diversos domínios dos SEE. Esta ferramenta, permite efetuar estudos de cenários mais realistas.

A Tabela 2.1 apresenta a perspetiva longitudinal do valor o produto.

Tabela 2.1: Proposta longitudinal de valor

Fase	Benefícios	Sacrifícios
Pré Compra	Permitir a interoperabilidade entre ferramentas; simular diversos cenários em simultâneo; distribuição automática por máquinas; simulação automática dos cenários; geração automática de resultados	Tempo
Compra	Nível relativo à definição de papéis e conteúdo, em relação ao domínio	Especificação dos modelos individuais e das interações
Pós Compra	Flexibilidade na composição de cenários; simulação distribuída de modo a agilizar o processo; diminuição do risco; simulação automática de várias ferramentas; comparação automática de resultados; sugestão de cenários que melhor satisfazem as suas necessidades;	Configuração dos cenários
Após Utilização	Satisfação; Solução para o problema pretendido	

2.2.5 Modelo de Negócio

A Figura 2.2 apresenta o modelo canvas relativo à solução proposta. A proposta desta dissertação está contextualizada no trabalho desenvolvido pelo GECAD, que é uma unidade de I&D.

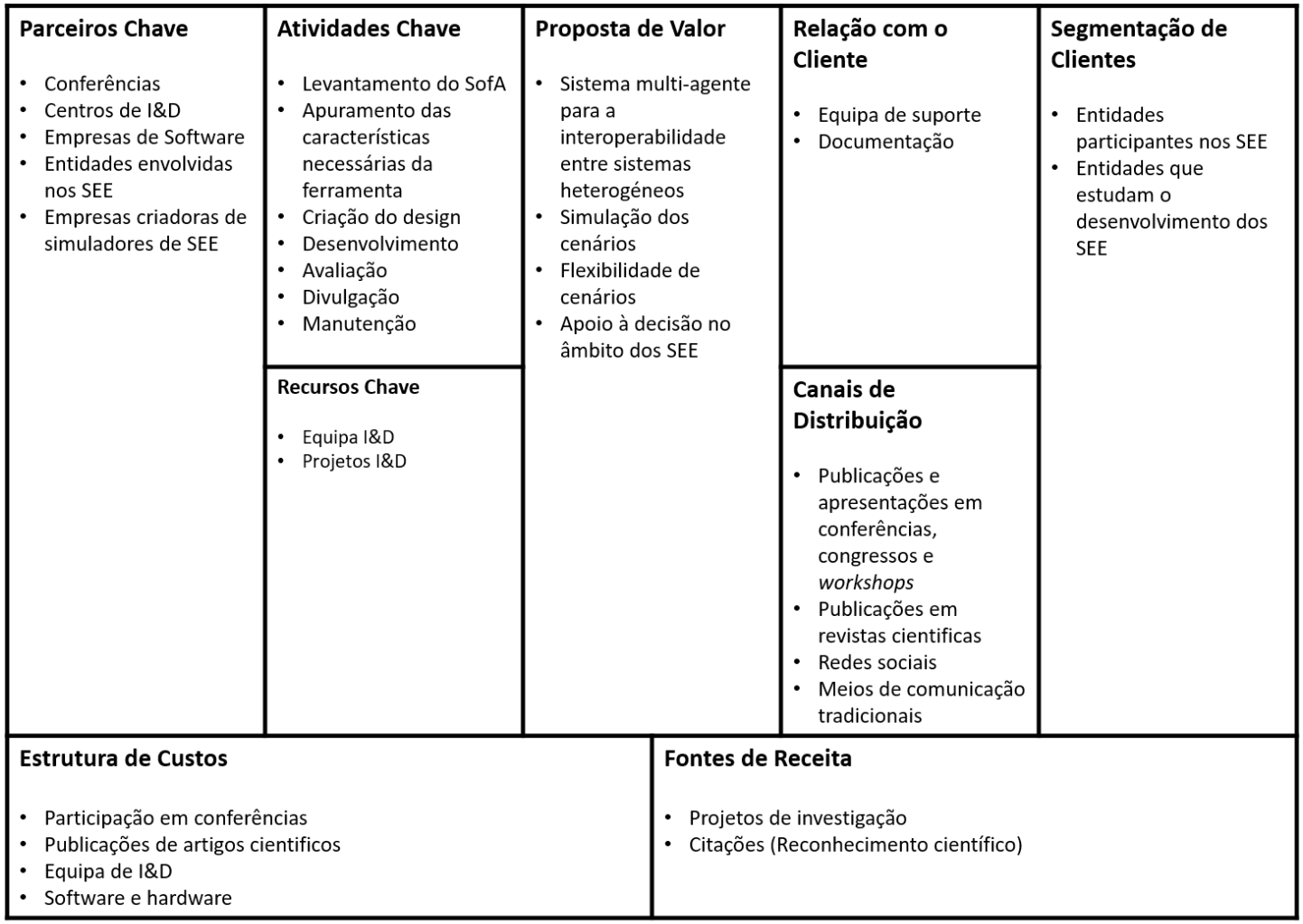

Figura 2.2: Modelo canvas

Na segmentação de clientes podemos identificar duas entidades como público alvo: entidades participantes nos SEE e entidades que estudam o desenvolvimento dos SEE. As entidades participantes podem ser produtores, consumidores, gestores da rede elétrica, entre outras entidades, que podem tirar benefício da ferramenta para o apoio à decisão. As entidades que pretendem estudar os SEE podem efetuar estudos sobre o comportamento futuro do setor, das entidades que o envolve, criação de novos modelos, entre outros.

A relação com o cliente é estabelecida através da equipa de suporte à ferramenta e pela documentação disponibilizada que demonstra como utilizar a ferramenta.

Os canais de distribuição são comuns no âmbito da investigação científica, nomeadamente através de publicações científicas; apresentações em conferências, congressos, entre outros; parcerias em projetos de I&D; redes sociais; e meios de comunicação tradicionais.

A proposta de valor consiste na oferta de uma ferramenta de simulação multi-agente para promover a interoperabilidade entre sistemas heterogéneos. Para tal, é necessário a criação de cenários flexíveis. Para além disso, também é pretendido que seja capaz de fornecer apoio à decisão aos seus utilizadores.

As atividades chave consistem: (i) no apuramento de ferramentas no estado de arte, uma vez que esta é uma preocupação recente, estão a surgir cada vez mais soluções que pretendem efetuar trabalho no mesmo sentido; (ii) no apuramento das características que a ferramenta deve ter, quer para a criação de cenários como para a análise de resultados; (iii) na criação do *design* da ferramenta; (iv) na avaliação dos resultados e refinação; (v) na divulgação ao publicar mais artigos científicos; e (vi) na manutenção, ao incluir novas ferramentas, modelos, entre outros.

Os parceiros chave influenciam o desenvolvimento da solução. Empresas criadoras de simuladores podem querer utilizar e incorporar as suas soluções nesta ferramenta, assim como outros centros de I&D; as entidades envolvidas nos SEE podem fornecer conhecimento e sugestões de melhorias à ferramenta; e as empresas de software podem compartilhar a experiência relativa às necessidades reais dos consumidores.

Os custos estão relacionados com as deslocações a conferências, publicações científicas, pagamento à equipa de desenvolvimento, compra de hardware como servidores e licenças de software.

As fontes de receita provêm de angariação de mais projetos de investigação e, numa perspetiva de reconhecimento que é importante na área, citações das publicações sobre o trabalho efetuado.

2.2.6 AHP

O *Analytic Hierarchy Process* (AHP) é um método de avaliação multi-critério desenvolvido por (Saaty 1990), que decompõe o problema numa estrutura hierárquica de níveis. Estes níveis são compostos pelos objetivos, critérios e alternativas. Adicionalmente, a cada critério é atribuído um peso, e comparado com os restantes através da Escala Fundamental. No final, é obtida a alternativa com melhor pontuação.

No contexto desta dissertação, o método AHP será utilizado na escolha da linguagem de comunicação entre os sistemas, sendo esta uma parte essencial para o sistema. Os critérios definidos são:

- Legibilidade: capacidade de perceção do conteúdo da mensagem;
- Complexidade: custo (esforço) na criação de uma mensagem;
- Performance: tempo despendido na interpretação da mensagem;
- Utilização: se é vulgarmente utilizado.

Após definidos os critérios, o passo seguinte será atribuir um peso de acordo com a importância do critério. O critério com maior importância será aquele que tiver um peso mais elevado.

Na definição de alternativas, no contexto desta dissertação podem ser identificadas as seguintes:

- *JavaScript Object Notation* (JSON);
- Resource Description Framework (RDF);
- Turtle.

Na construção do modelo é necessário comparar todos os critérios entre si, atribuindo valores entre 1 e 9, onde 1 significa que os critérios possuem igual importância, e 9 refere-se à importância absoluta em relação ao outro. Esta comparação traduz-se numa matriz correspondente à Prioridade Relativa dos critérios. A Prioridade Relativa é obtida através da média aritmética normalizada, de cada linha da tabela.

A Tabela 2.2 apresenta a avaliação para o critério Legibilidade.

Tabela 2.2: Avaliação para o critério Legibilidade.

Legibilidade	**JSON**	**RDF**	**Turtle**	**Prioridade Relativa**
JSON	1	4	6	0,5339
RDF	1/4	1	2	0,1520
Turtle	1/6	1/2	1	0,0839

Para o critério Legibilidade a alternativa com menor peso é a Turtle. Isto porque, além de extenso, as propriedades inerentes da linguagem utilizam uma versão abreviada, que pode dificultar a leitura da definição dos conceitos e das suas relações. Por outro lado, no JSON esta relação é evidente pela sua estrutura hierárquica e pelas chaves dos objetos.

A Tabela 2.3 apresenta a avaliação para o critério Complexidade.

Tabela 2.3: Avaliação para o critério Complexidade.

Complexidade	**JSON**	**RDF**	**Turtle**	**Prioridade Relativa**
JSON	1	4	4	0,6667
RDF	1/4	1	1	0,1667
Turtle	1/4	1	1	0,1667

Para o critério Complexidade é possível observar que tanto o RDF como o Turtle possuem pesos iguais. Isto acontece porque eles são desenvolvidos de igual forma. Contudo, a

criação de uma mensagem em linguagem JSON é muito mais simples, devido também à sua estrutura simplificada.

A Tabela 2.4 apresenta a avaliação para o critério Performance.

Tabela 2.4: Avaliação para o critério Performance.

Performance	JSON	RDF	Turtle	Prioridade Relativa
JSON	1	6	4	0,6853
RDF	1/6	1	1/3	0,0934
Turtle	1/4	3	1	0,2213

É possível observar através da Tabela 2.4 que a melhor alternativa para o critério Performance é o JSON novamente. Além de possuir muitas vezes um tamanho de ficheiro mais reduzido devido à simplicidade da sua sintaxe, estão disponíveis diversas ferramentas de interpretação desta linguagem devido à sua popularidade, com um elevado grau de performance.

A Tabela 2.5 apresenta a avaliação para o critério Utilização.

Tabela 2.5: Avaliação para o critério Utilização.

Utilização	JSON	RDF	Turtle	Prioridade Relativa
JSON	1	8	8	0,7898
RDF	1/8	1	1/2	0,0812
Turtle	1/8	2	1	0,1290

Por último, no critério Utilização a hipótese que mais se destaca é novamente o JSON, uma vez que é a alternativa com utilização mais comum. Sendo o objetivo a integração com sistemas externos, é importante que a linguagem escolhida seja acessível para o desenvolvimento de parceiros e outras entidades, que pretendam comunicar com o sistema proposto.

Para conclusão, de forma de apurar qual das alternativas é a mais indicada para o problema é necessário proceder ao cálculo da Prioridade Composta. Este é feito através do somatório da multiplicação da avaliação efetuada de cada alternativa, em cada critério, pelo peso do critério. Os valores obtidos estão presentes na Tabela seguinte:

Tabela 2.6: Prioridade Composta das alternativas.

	Legibilidade	Complexidade	Performance	Utilização	Prioridade Composta
JSON	0,5339	0,6667	0,6853	0,7898	0,6621
RDF	0,1667	0,0934	0,1667	0,0812	0,1474
Turtle	0,0839	0,1667	0,2213	0,1290	0,1640

Analisando a Tabela a cima conclui-se que a alternativa mais indicada para o problema é a utilização da linguagem JSON, sendo que é a que possui valor mais elevado. As outras duas

alternativas apresentaram resultados semelhantes, contudo, bastante abaixo da alternativa vencedora.

2.3 Agentes e Sistemas Multi-Agente

Deste o surgimento da Inteligência Artificial (IA) (Turing 1950) que o seu principal foco foi desenvolver sistemas computacionais capazes de solucionar problemas de forma equivalente ao pensamento humano, apresentando inteligência, autonomia, raciocínio e capacidade de aprendizagem (McCarthy et al. 2006).

Foi na década de 80 que surgiu o ramo da inteligência artificial distribuída, resultante da união entre a IA com os sistemas distribuídos. Este novo ramo, segundo Davis 1980, veio solucionar problemas onde uma única entidade dotada de IA não consegue fornecer a resposta mais adequada. Desta forma, os sistemas inteligentes estão inseridos num ambiente distribuído - sistema distribuído - que lhe confere um caráter mais eficiente e melhor capacidade de processamento. A inteligência artificial distribuída divide-se em duas áreas: resolução distribuída de problemas e SMA (Panait e Luke 2005). A resolução distribuída de problemas consiste na segmentação de um problema por vários nós, resultando em problemas com menor complexidade. Com este método, cada nó é responsável por obter a solução para o seu segmento, contribuindo para uma solução coletiva. A área SMA centra-se no comportamento de entidades inteligentes – agentes – que possuem um determinado grau de autonomia e complexidade resultante do seu número de interações. O conceito da inteligência artificial distribuída veio dar um contributo imprescindível à IA, uma vez que o sistema distribuído permite tornar a análise do problema mais intuitiva e simplificada, além de dotar o sistema de uma maior tolerância a falhas do sistema (Ferber 2001).

2.3.1 Agente de Software

Na IA a definição do termo agente não é consensual (Huang et al. 2010; Minsky e Marvin 1986; Schleiffer 2005). Contudo, podem ser identificadas características que são consideradas comuns a várias definições, tais como a autonomia, capacidade social com outros agentes, sensorização do ambiente envolvente e reatividade às alterações desse ambiente (Weiss 1999).

Segundo Luck e D'Inverno 1995; Wooldridge e Nicholas R Jennings 1995, um agente de *software* é um sistema computacional que está inserido num determinado ambiente, onde, de forma autónoma, é capaz de reagir a estímulos provenientes desse ambiente e tomar decisões de forma a alcançar os seus objetivos. De forma a atingir esses objetivos Weiss 2000 propôs uma lista de características inerentes ao agente:

- Capacidade Sensorial: sensibilidade de perceção do ambiente onde se insere;
- Reatividade: capacidade de reação e adaptação conforme as alterações do ambiente.
- Autonomia: capacidade de controlo das suas próprias ações;
- Pro-Atividade: capacidade de se direcionar a um objetivo;
- Persistência: quanto um agente existe durante longos períodos de tempo;
- Capacidade Social: habilidade de comunicar com outros agentes do sistema;

- Aprendizagem: adquirir experiência com o resultado das suas ações e adaptar as decisões futuras;
- Mobilidade: habilidade de transitar entre máquinas consoante a perceção do ambiente;
- Flexibilidade: caso as suas tarefas não necessitem de ser pré-determinadas;
- Agilidade: analisar e identificar novas oportunidades;
- Caráter: apresentar uma determinada personalidade;
- Inteligência: efetuar a ligação entre as características da aprendizagem e autonomia.

Para além destas, outros autores (Cui-Mei 2009; Huang et al. 2010) ainda identificam características adicionais, como por exemplo:

- Personalização: capacidade de representação da informação e comportamento de uma dada entidade;
- Racionalidade: maximização das suas conquistas, conjugando o cumprimentos dos seus próprios objetivos com sucesso;
- Veracidade (Honestidade): o agente não pode transmitir informação falsa;
- Sanidade: a concretização de tarefas deve ser apenas em prol dos seus objetivos, e não de forma cega.

2.3.2 Sistemas Multi-Agente

Stone e Veloso 2000 dividem os sistemas baseados em agentes em dois tipos: sistemas com agente único e SMA. Os SMA são sistemas computacionais que possuem a diversos agentes que interagem entre si com a finalidade de resolver um problema. Essa interação resulta da necessidade de utilizar capacidades ou conhecimento que um único agente não possui, necessitando de recorrer a outro para a resolução de uma tarefa mais complexa (Panait e Luke 2005). Estes sistemas, comparativamente aos sistemas com apenas um agente, permitem mais confiabilidade, robustez, modularidade, escalabilidade, adaptabilidade, concorrência, paralelismo e dinamismo (Elamy 2005).

Portanto, cada agente é autónomo e pode ter uma arquitetura heterogénea dos demais (Nicholas R. Jennings, Sycara e Wooldridge 1998). Este tipo de sistemas tem características muito específicas:

- Cada agente possui um ponto de vista limitado;
- Não existe controlo completo do sistema;
- Os dados são descentralizados;
- A computação é assíncrona.

Nos últimos anos, o interesse por estes sistemas tem sido cada vez maior pela sua grande capacidade de resolução de problemas complexos e altamente dinâmicos. Por isso é utilizado em áreas como comércio eletrónico, sistemas de produção, controlo de tráfego, filtragem de informação, obtenção e classificação de informação, aplicações aeroespaciais e MEE.

Por outro lado, os SMA também são uma ferramenta utilizada para a melhoria do desenvolvimento de *software* devido ao aumento da capacidade de processamento, abstração do problema, interligação com outros sistemas, prevenção de falhas e simplicidade conceptual.

2.3.3 Padrões

Com a utilização dos Sistemas Baseados em Agentes (SBA) em diversas áreas de aplicação, que exigem diferentes níveis de complexidade, características como a flexibilidade, robustez e extensibilidade destes sistemas ganham especial importância (McArthur et al. 2007a). Para tal, torna-se primordial a utilização de ferramentas padronizadas para o desenvolvimento de SBA, quer na componente arquitetural dos agentes, como nas suas comunicações (McArthur et al. 2007b).

A *Foundation for Intelligent Physical Agents* (FIPA)[1] é uma organização da *IEEE Computer Society* que foi oficialmente aceite em 2005, com o objetivo de disponibilizar padrões para a implementação de SBA, promovendo a interoperabilidade e reutilização. Para tal, a FIPA desenvolveu padrões e modelos descritivos para as perspetivas do desenvolvimento de SBA (Poslad e Charlton 2001), nomeadamente funcional, informacional, operacional e organizacional. O modelo funcional permite a definição das propriedades e comportamentos dos agentes através do uso de descrições informais, podendo considerar regras de pré e pós condições que definem as regras do comportamento; o modelo informacional prende-se com a estrutura da troca de informação e persistência; o modelo operacional (ou de interação) está associado à sequencia e interação que os comportamentos dos agentes deverão respeitar para o funcionamento normal do sistema; e, por fim, o modelo organizacional define a estrutura dos agentes, qual é a relação entre os diferentes tipos de agentes e a sua possível formação de grupos.

A Figura 2.3 apresenta o modelo de referência FIPA para a gestão de agentes.

Por observação à Figura 2.3, é possível identificar os seguintes elementos:

- Facilitador de diretório (*Directory Facilitator (DF)*): é um componente opcional da Plataforma de Agentes (*Agent Plataform* (AP)) e deve ser implementada como um serviço de páginas amarelas dos serviços disponibilizados pelos agentes.
- Agente gestor do sistema (*Agent Management System (AMS)*): é um componente obrigatório e único da AP, que efetua o controlo supervisionado da mesma. Mantém o diretório do *Agent Identifier* (AID) de todos os agentes acessíveis e registados na AP. Para além disso, oferece um serviço de páginas brancas a outros agentes e é responsável por garantir um AID único a cada agente, através do seu registo.
- Serviço de transporte de mensagens *(Message Transport System (MTS)*): é o método de comunicação utilizado entre os agentes.
- Plataforma de Agentes (*AP*): fornece a estrutura física onde os agentes se encontram. Está interligada à(s) máquina(s), sistema operativo, software de suporte aos agentes, componentes (DF, AMS e MTS) e agentes.
- Software: descreve todas as instruções executadas por um agente, que não provêm de outro agente.

[1] FIPA - `http://www.fipa.org/index.html`

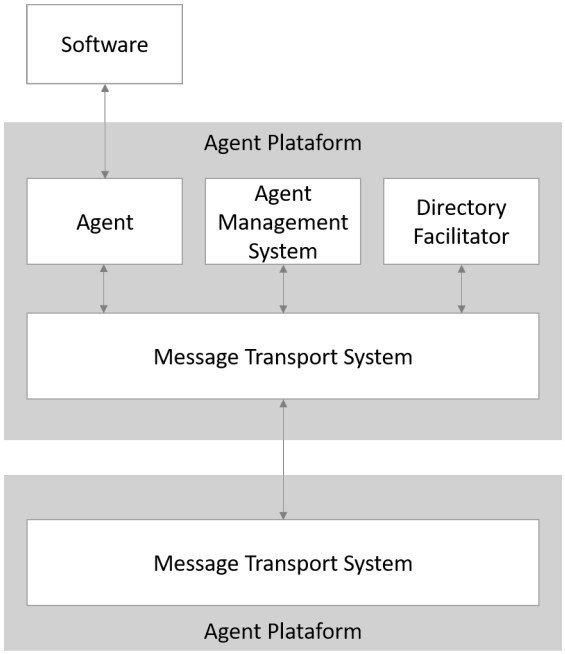

Figura 2.3: Modelo de referência para gestão de agentes FIPA (FIPA 2004)

- Agente (*Agent*): é o ator principal da AP, e possui um ou mais serviços, que são publicados no DF. O Agente é identificado pelo AID que o rotula como uma entidade única no universo de agentes. Esta entidade comunica através de linguagens de comunicação padronizadas (secção 2.4.2).

2.4 Comunicação e Interoperabilidade entre Agentes

A comunicação é a base dos SBA. É através da troca de informação, conhecimento e serviços que os agentes estão aptos a atingir os seus objetivos e solucionar problemas (Fornara e Gambardella 2003). Podemos considerar que existem os seguintes tipos de comunicação (Huhns e Stephens 1999):

- Comunicação direta: consiste num troca bilateral de mensagens. Para isso, o agente que quer estabelecer a comunicação necessita conhecer qual é o serviço que o outro agente disponibiliza, como a comunicação pode ser estabelecida e que informação é necessária. No entanto, quando os agentes enviam mensagens em simultâneo e reciprocamente pode ocorrer o bloqueio do sistema. Este facto deve-se à inexistência de um elemento capaz de coordenar as comunicações.
- Comunicação assistida: serve-se de entidades específicas – agentes facilitadores e *brokers* – que são responsáveis por distribuir/reencaminhar as mensagens para os agentes respetivos. Assim, os agentes não necessitam de conhecer aqueles com os quais deseja comunicar, mas o facilitador necessita de conhecer todos os agentes e as suas características. Este tipo de arquitetura tem um problema comum: estrangulamento de mensagens. O sistema fica dependente do facilitador para poder prosseguir com as comunicações.

- Comunicação em *broadcast*: é um tipo de comunicação indireta, em que o agente envia a mensagem para toda a comunidade de agentes. Este tipo de comunicação é útil quando o agente em questão não sabe a quem deve dirigir a sua mensagem, ou quando quer notificar a comunidade de algo. A desvantagem reside no facto de não ser garantido que o(s) agente(s) de destino receba(m) a mensagem de facto.
- Comunicação através de *blackboard*: é um tipo de comunicação indireta, onde os agentes comunicam através de um segmento de memória partilhada e qualquer agente que tem acesso pode responder aos pedidos lá colocados. Esta solução pode ser comparada a um *brainstorming* entre agentes (Engelmore e Morgan 1988).

Contudo, a transmissão de uma mensagem não é o suficiente para os agentes serem capazes de interoperar. A comunicação é caracterizada não só pela passagem de informação, mas também pela partilha do conhecimento e coordenação com o outro. Assim, para tornar uma comunicação verdadeiramente eficaz é necessário estabelecer padrões que considerem: (i) o protocolo de comunicação; (ii) as linguagens de comunicação (sintaxe); e (iii) semântica que represente o conteúdo da mensagem.

2.4.1 Protocolos de Comunicação

Um protocolo de interação é um conjunto de regras que descrevem a comunicação entre agentes, permitindo definir como estes devem lidar com a receção das novas mensagens (Marzougui e Barkaoui 2013). Quando um agente utiliza determinado protocolo, este terá de assegurar também a sintaxe e semântica que lhe são associados. O estilo de interação a aplicar vai depender do tipo de comunicação entre os agentes. No tipo de comunicação direta, normalmente é usado a Linguagem de Comunicação de Agentes (*Agent Communication Language* (ACL)) (ver secção 2.4.2), enquanto que na conversação indireta é possível identificar os seguintes protocolos:

- Protocolo de coordenação: executado entre agentes, para satisfazer os seus objetivos individuais ou do grupo. Os objetivos podem sofrer ajustes ou deixar de existir, contudo, os outros agentes devem ser informados dessas alterações.
- Protocolo de cooperação: consiste na decomposição de uma tarefa em sub-tarefas, e distribui-las pelos agentes da comunidade. Este mecanismo permite lidar com tarefas de elevada complexidade.
- Protocolo de negociação: usado em casos em que os agentes possuem diferentes objetivos perante o mesmo recurso ou tarefa. Para tal, os agentes devem negociar para tomar uma decisão final.

A FIPA disponibiliza diversos protocolos padrão, de acordo com situações bem definidas. Alguns exemplos são o *FIPARequest* (FIPA 2002d), *FIPAQuery* (FIPA 2002c), *FIPAContractNet* (FIPA 2002b), *FIPAAuctionEnglish* (FIPA 2001b), *FIPABrokering* (FIPA 2001a), *FIPAPropose* (FIPA 2001c), entre outros.

2.4.2 Linguagens de Comunicação

A sintaxe[2] pode ser definida como a *"parte linguística que estuda e descreve as relações que as palavras estabelecem entre si numa frase"*, ou como o *"grupo de regras que organizam qualquer tipo de linguagem"*. Assim, de forma análoga, as linguagens de comunicação entre agentes (ACL) permitem definir a representação da organização do conteúdo e assentam sobre a utilização de performativas, cujo conceito é oriundo da teoria dos atos de fala (Fornara e Gambardella 2003). Entre as linguagens de comunicação mais conhecidas estão a *Knowledge Query Manipulation Language* (KQML) e FIPA ACL:

- KQML (Finin et al. 1994) foi desenvolvida pela DARPA Knowledge Sharing Effort, e consiste numa linguagem onde não há necessidade de conhecer o conteúdo, formato e conhecimento associado da mensagem. A única informação necessária, é o destinatário, que deverá saber exatamente o que esperar da mensagem trocada. A KQML possui algumas performativas, mas muitas vezes são resumidas ao tipo falar (*tell*) e perguntar (*query*).
- FIPA ACL é uma linguagem de comunicação padronizada criada pela FIPA. Uma das suas principais vantagens prende-se com o facto de permitir uma grande precisão na descrição do ato comunicativo, bem como os seus efeitos, através da semântica (Dignum e Greaves 2000). Além disto, a FIPA ACL possui um leque alargado de performativas (atos comunicativos), baseadas no conceito crenças-desejos-intenções (*belief-desire-intention*).

2.4.3 Ontologias

Para a comunicação efetiva de agentes que precisam efetuar trabalho colaborativo é necessário que estes estejam dotados de um modelo de socialização (Poslad e Charlton 2001). Para isso, os agentes necessitam de conhecer vocabulário comum e compreender a relação entre os conceitos definidos, de modo a extrair conhecimento e interpretar as mensagens trocadas, para desempenhar corretamente as suas tarefas. Na IA, a definição lógica e relação entre estes conceitos dá-se pelo nome de ontologia (Fornara e Gambardella 2003).

Uma ontologia especifica a representação de vocabulário para um determinado domínio, bem como as suas relações e funcionalidade. Na literatura, é possível encontrar diversas definições deste conceito, mas Thomas R Gruber 1993 descreve uma ontologia como sendo uma especificação e conceptualização do conhecimento, que uma comunidade de agentes pode usar na sua comunicação. Studer, Benjamins e Fensel 1998 detalha esta definição adicionando as seguintes características:

- Conceptibilidade: é o domínio abstrato e racional, incluindo a identificação e descrição de conceitos, propriedades e relações entre eles;
- Especificidade: é o detalhe, precisão, consistência e solidez da descrição do domínio;
- Explicitabilidade: prende-se com a representação da conceptualização de modo a que os agentes se possam compreender e processar a informação;
- Formalismo: pretende que máquinas e humanos possam ler, compreender e processar a ontologia;

[2]Significado de *sintaxe* - `https://www.infopedia.pt/dicionarios/lingua-portuguesa/sintaxe`

- Partilha: implica que uma ontologia é aceite por toda a sociedade de agentes.

As ontologias permitem a representação abstrata e organizada de um domínio, fornecendo uma compreensão comum da estrutura da informação para humanos e máquinas (Hepp 2008). Thomas R. Gruber 1995 identifica vários benefícios oriundos da utilização das ontologias: (i) entendimento comum do domínio de conhecimento entre as entidades que possuem um objetivo comum; (ii) partilha do domínio de conhecimento; (iii) reutilização do domínio de conhecimento;(iv) análise do domínio de conhecimento; (v) separação do domínio de conhecimento do conhecimento operacional; e (vi) tornar assunções do domínio explicitas.

Uma das características mais importantes da implementação de uma ontologia prende-se com o facto de que para esta ser utilizada por todos os agentes é necessário que eles entrem em total acordo acerca da definição dos termos que a ontologia contém.

Gruninger e Lee 2002 defende que as ontologias servem três propósitos: (i) comunicação entre sistemas computacionais, comunicação entre humanos e comunicação entre sistemas computacionais e humanos; (ii) inferência computacional para a representação interna e manipulação de informação, bem como análise de estruturas, algoritmos, dados e entrada e saída dos sistemas implementados, sob a forma teórica e conceptual; (iii) e reutilização do conhecimento para a estruturação e organização de bibliotecas ou repositórios para planeamento e informação de domínio.

Para os agentes utilizarem uma ontologia é necessário que todos concordem com a definição dos conceitos que ela contém, bem como a sua representação. No entanto, é possível que um agente possa usar mais do que uma ontologia, onde a utilização de cada uma delas permite a comunicação com um determinado agente (Hadzic et al. 2009).

Segundo Hadzic et al. 2009, o mesmo conhecimento pode ser expresso de diferentes formas, tendo como base os seguintes parâmetros: grau de formalidade; grau de granularidade; nível de generalidade; quantidade, tipo e assunto de conceptualização; e expressividade. Na lista que se segue, é apresentado mais detalhe sobre cada um destes parâmetros:

- Grau de Formalidade (Figura 2.4)
  - Ontologias informais (*highly informal ontologies*): são expressas em linguagem natural, e devido a isso, possui definição de termos de forma ambígua, dificultando a interpretação dos conceitos por parte dos agentes.
  - Ontologias semi-informais (*semi-informal ontologies*): são expressas em linguagem natural, mas de forma estruturada e restrita, ajudando a reduzir a ambiguidade e promovendo a clareza.
  - Ontologias semi-formais (*semi-formal ontologies*): são expressas em linguagens artificiais.
  - Ontologias formais (*rigorously formal ontologies*): são expressas por termos que são definidos de modo preciso, através de semântica e teoremas formais.
- Grau de Granularidade
  - Granularidade grosseira (*coarse ontologies*): ontologia partilhada com entidades que concordaram com a conceptualização do domínio. Possui um número mínimo de axiomas escritos com o mínimo de expressividade.

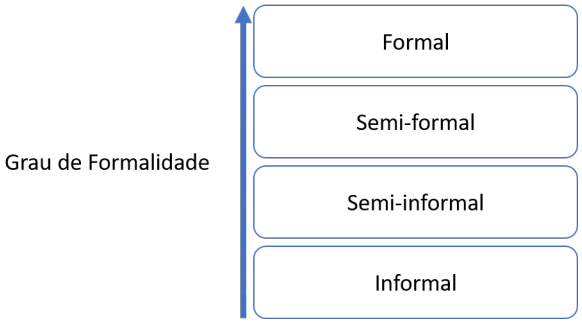

Figura 2.4: Classificação das ontologias de acordo com o grau de formalidade (adaptado de Hadzic et al. 2009)

  - Granularidade fina (*fine-grained ontologies*): ontologia que é usada para estabelecer um acordo da conceptualização do domínio entre as entidades. Esta possui uma grande quantidade de axiomas escritos numa linguagem com um elevado grau de expressividade.

- Nível de Generalidade (Figura 2.5)
  - Ontologias de aplicação (*application ontologies*): o vocabulário é relativo a um determinado domínio e tarefa. São especializações de ontologias de tarefa e ontologias de domínio.
  - Ontologias de tarefa (*task ontologies*): o vocabulário é relativo a uma determinada tarefa ou atividade.
  - Ontologias de domínio (*domain ontologies*): o vocabulário é relativo a um determinado domínio.
  - Ontologias de alto-nível (*top-level ontologies*): conceitos gerais ou conhecimento baseado em senso comum. São independentes do problema e domínio.

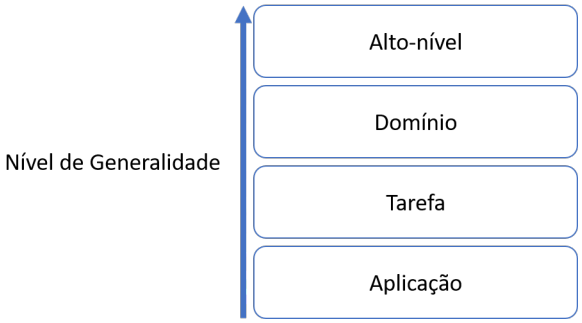

Figura 2.5: Classificação das ontologias de acordo com o nível de generalidade (adaptado de Hadzic et al. 2009)

- Quantidade, tipo e assunto da conceptualização
  - Estrutura da conceptualização (Figura 2.6)
    - Ontologias terminológicas (*terminological ontologies*): léxico usado para especificar terminologia para representação do conhecimento relativo a um domínio. Não captura a semântica dos termos.

* Ontologias de informação (*information ontologies*): especificam a estrutura de registo de bases de dados para permitir o seu controlo e gestão. Não definem conceitos que são instanciados pelas instâncias de bases de dados.
* Ontologias de modelação de conhecimento (*knowledge modeling ontologies*): especificam a conceptualização do conhecimento. Possuem uma estrutura mais completa comparativamente às ontologias de informação.

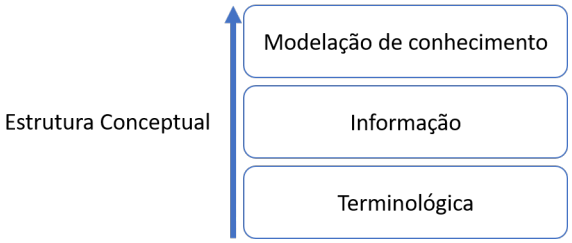

Figura 2.6: Classificação das ontologias de acordo com a estrutura de conceptualização (adaptado de Hadzic et al. 2009)

– Classificação do assunto
  * Ontologias de aplicação (*application ontologies*): contêm os conceitos necessários para a modelação do conhecimento necessário para especificar as aplicações. Servem-se de termos de ontologias mais genéricas e estendem o seu conhecimento através da representação de componentes específicos de métodos e tarefas.
  * Ontologias de representação (*representation ontologies*): fornecem uma *framework* de representação de ontologias de domínio e genéricas, que podem ser descritas através do uso de primitivas fornecidas em ontologias deste tipo.
  * Ontologias de domínio (*domain ontologies*): contêm conceitos específicos para um determinado domínio. Especificam restrições a serem aplicadas à estrutura e conteúdo de um domínio de conhecimento. O seu desenvolvimento depende do conhecimento.
  * Ontologias genéricas (*generic ontologies*): especificam conceitos que são comuns a vários setores. A especialização das ontologias genéricas resultam em ontologias de domínio;

- Expressividade (Figura 2.7)
  – Vocabulários controlados (*controlled vocabularies*): versão mais simples de ontologia. Contém um número finito de termos com uma interpretação ambígua.
  – Glossários (*glossaries*): Lista de termos, cujos significados são definidos. Geralmente, são expressos em linguagem natural e são destinados apenas a humanos, ou seja, não são processados por agentes de *software*.
  – Thesaurus (*thesauri*): define as relações entre os termos e adiciona semântica aos glossários. Não fornece uma estrutura hierárquica explicita. Um agente pode interpretar as relações definidas por este tipo de expressividade.

- Hierarquia informal "é-um"(*informal is-a hierarchies*): fornece uma noção de generalidade e especificidade. A herança não pode ser assumida devido ao conceito informal de "é-um". Uma instância de uma classe mais específica não é necessariamente uma instância de uma classe mais generalizada.
- Hierarquia formal "é-um"(*formal is-a hierarchies*): ontologias onde os conceitos formam uma hierarquia rígida de subclasses, e a herança é sempre aplicável.
- Relações de instâncias formais (*formal instances relations ontologies*): possuem uma estrutura hierárquica. As relações da instância formal são válidas quando o conteúdo da ontologia descreve os indivíduos e as suas relações, considerando os conceitos que elas instanciam.
- *Frames* (*frames ontologies*): descrevem conceitos agregados às suas propriedades. Pode ser aplicada herança às propriedades. Um conceito específico pode herdar propriedades de outro mais generalizado.
- Restrição de valor (*value restriction ontologies*): aplicação de restrições aos valores associados com propriedades. Normalmente, estas restrições são herdadas por conceitos mais específicos.
- Restrições lógicas gerais (*general logical constraints ontologies*): ontologias que foram escritas com linguagens que possuem um grau de expressividade elevado, resultando numa especificação completa dos conceitos e das suas propriedades. De todos os tipos, estas ontologias possuem o maior grau de expressividade.

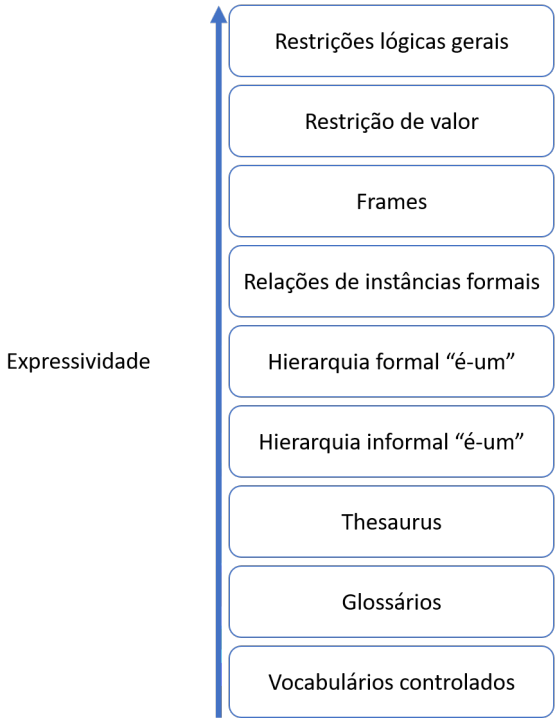

Figura 2.7: Classificação das ontologias com base na sua expressividade (adaptado de Hadzic et al. 2009)

2.4.4 Linguagens de Representação de Conhecimento

O conteúdo transmitido numa mensagem envolve a linguagem de conteúdo e ontologia, sendo que a linguagem fornece a sintaxe do conteúdo, e a ontologia o léxico que ele contém (McArthur et al. 2007b). A FIPA, de modo a representar o conteúdo das mensagens utilizado pelo padrão de comunicação FIPA ACL, definiu linguagens de conteúdo de forma a expressar a formalidade, objetos, proposições, relações e funções (Fornara e Gambardella 2003). Estas linguagens de conteúdo são:

- FIPA Semantic Language (SL) (FIPA 2002e): expressa-se em três vertentes: (i) a proposição, que possui uma fórmula bem definida à qual pode ser atribuído um valor de verdade num determinado contexto; (ii) a ação, que pode realizar-se em sequência ou paralelamente a outras ações; e (iii) o *Identifying Reference Expression* (IRE), que identifica o objeto num determinado domínio.
- FIPA Constraint Choice Language (CCL) (FIPA 2002a): baseia-se na representação de problemas relacionados com múltiplas opções inter-relacionadas. Permite suportar problemas baseados em satisfação de restrições.
- *Knowledge Interchange Format* (KIF) (Genesereth e Fikes 1992): possui uma semântica declarativa e destina-se a descrever fatos sobre o mundo em vez de processos ou procedimentos, podendo servir-se de funções, relações e regras. A sua lógica é de primeira ordem, e por isso fornece a expressão arbitrária no cálculo de predicados.
- RDF (W3C 2004): é uma linguagem cujas vantagens são caracterizadas pela sua extensibilidade, reutilização, simplicidade e por ser um padrão para aplicações web. A sua sintaxe pode ser codificada em *eXtensible Markup Language* (XML), Turtle, N3, Ntriple, Json-LD, entre outros.

2.5 Plataformas de Desenvolvimento de Sistemas Multi-agente

Na literatura existe um grande número de plataformas destinadas ao desenvolvimento de SMA. Por vezes, a implementação deste tipo de sistemas pode ser bastante complexa, nomeadamente no desenho da arquitetura dos sistemas, dos modelos dos agentes e das suas comunicações. As plataformas de desenvolvimento, permitem simplificar este processo, através da disponibilização de serviços que efetuam a gestão do ciclo de vida dos agentes, da comunicação entre eles, e da perceção dos agentes perante o ambiente (Rousset et al. 2014).

Algumas das plataformas mais conhecidas são:

- FIPA-*Open Source* (OS) (Poslad, Buckle e Hadingham 2000): é uma plataforma de agentes aberta, uma vez que assenta em software de distribuição e *schema open-source*. A ferramenta inclui: diferentes tipos de agentes; diferentes camadas para comunicação entre agentes; gestão das mensagens e diálogos entre agentes; plataforma dinâmica que suporta vários protocolos de transporte de mensagens; interfaces abstratas para padrões; e ferramentas de diagnóstico e visualização. As linguagens utilizadas por esta ferramenta também usam os padrões definidos pela FIPA, nomeadamente a FIPA ACL para a linguagem de comunicação, e FIPA SL, FIPA CCL e FIPA RDF como linguagens de conteúdo.

- *Java Agent DEvelopment Framework* (JADE) (Bellifemine et al. 2007): é uma ferramenta implementada em Java e *open-source*, que segue os padrões criados pela FIPA, permitindo a interoperabilidade entre sistemas heterogéneos. O JADE fornece serviços para a criação de agentes, compreendendo os seus comportamentos e comunicações segundo a linguagem de comunicação FIPA ACL, além de disponibilizar ferramentas gráficas para acompanhamento das comunicações. Permite serviços de *debbugging*, pesquisa de serviços, mobilidade de agentes, entre outros. Para a linguagem de representação de conteúdo, a ferramenta utiliza por defeito o FIPA SL, mas permite a utilização de outras linguagens através da serialização e desserialização de dados.
- *Open Agent Architecture* (OAA) (Martin, Cheyer e Moran 1999): foca-se na construção de uma comunidade distribuída de agentes. Esta ferramenta tem como principal característica a utilização do registo das tarefas a executar num *blackboard*, onde o agente facilitador coordena os agentes do sistema que vão executar essas tarefas consoante as suas capacidades. É possível ter mais do que um agente facilitador por sistema. Possui três tipos de agentes: aplicacionais, que estabelecem ligação com *Application Programming Interface* (API); meta-agentes que servem-se de conhecimento e raciocínio para coordenar agentes consoante regras; e de interface com o utilizador, que recolhem dados provenientes do utilizador, como gestos, discurso, etc. Os agentes estabelecem as suas comunicações através da linguagem *Interagent Communication Language* (ICL).
- ZEUS (Collis et al. 1998): permite a elaboração de SMA colaborativos e utiliza a linguagem de programação Java, tirando proveito da sua natureza orientada a objetos. A ferramenta ZEUS possui três componentes: a biblioteca de componentes do agente, que permite a implementação das funcionalidades necessárias para a criação de um agente colaborativo (comunicação assíncrona, *mailbox*, ontologias, planeamento de tarefas, etc); ferramentas de visualização para *debugging* de agentes; e software de construção de agentes, que respeita uma metodologia de desenvolvimento bem definida, que aborda definição do domínio, agentes, a sua organização, coordenação, tarefas, regras e definições de factos. A ferramenta também segue os padrões da FIPA e utiliza as linguagens KQML e FIPA SL.

2.6 Simulação em Sistemas Elétricos de Energia

Num ambiente tão complexo e multidisciplinar como os SEE, o uso de ferramentas de simulação para o estudo dos seus subsistemas torna-se imprescindível (Haan et al. 2011a). O uso de simulação traz vários benefícios ao setor. Numa perspetiva económica e social, a simulação permite reduzir custos relacionados com a atualização dos sistemas reais, bem como a previsão do comportamento futuro dos SEE, através do estudo de diferentes cenários. Por outro lado, do ponto de vista prático das entidades que constituem os SEE, através da simulação e estudo destes sistemas é possível entender como diferentes mecanismos de mercado podem afetar vendedores e compradores de energia; fazer uma previsão do consumo de energia numa rede elétrica e para um determinado instante, para planear a gestão de energia; explorar novas possibilidades para modelos de mercado alternativos; aprofundar a compreensão de conceitos relacionados com a DR no que diz respeito ao seu impacto nos consumidores; quais as tarifas de DR mais apropriadas para os perfis de consumidores; planeamento de distribuição da rede; entre outros.

Para tornar a simulação dos SEE possível é necessário considerar alguns aspetos importantes. O primeiro relaciona-se com a modelação dos sistemas, que devem cobrir as entidades participantes, o comportamento, os recursos, a rede, e outros conteúdos importantes, necessários para a representação fiel dos sistemas reais. Outro aspeto prende-se com a flexibilidade dos cenários de simulação (Schutte, Scherfke e Troschel 2011), que devem permitir lidar com a complexidade do sistema, bem como a escalabilidade. A resolução temporal também é um tópico chave, especialmente na interoperabilidade entre sistemas (Georg et al. 2012; Schutte, Scherfke e Troschel 2011), uma vez que é necessário proceder à sua sincronização. E finalmente, questões menores também devem ser consideradas, nomeadamente a adaptabilidade do simulador ao utilizador, onde este pode personalizar os modelos disponíveis; a independência do sistema relativamente a uma plataforma ou linguagem de programação; simulação em tempo real; entre outros.

2.6.1 Ferramentas de Simulação

É cada vez mais comum desenhar SEE sob uma arquitetura multi-agente, uma vez que a sua natureza distribuída permite lidar com a complexidade do setor (Howell et al. 2017). Além disso, cada agente pode representar uma entidade ou ponto de vista do sistema, que possui os seus próprios desejos, restrições, autonomia e flexibilidade. Desta forma, um agente pode representar um participante do mercado, um componente de rede ou um sistema.

Segue-se uma lista de simuladores multi-agente existentes na literatura:

- ALBidS (Pinto et al. 2014): é uma ferramenta multi-agente que fornece apoio à decisão a entidades que desejam participar nos MEE com o modelo em bolsa, nomeadamente com o papel de comprador ou vendedor. Este sistema utiliza técnicas de inteligência artificial para determinar quais são as melhores propostas no mercado, dependendo do objetivo da entidade. Além disto, o simulador possui diversos perfis de entidades e adota a estratégia mais apropriada dependendo do oponente. Esta ferramenta pode interligar-se diretamente com o simulador MASCEM, onde as entidades participam no mercado adotando a estratégia fornecida pelo ALBidS. O JADE é usado para o seu desenvolvimento.
- MASCEM (Santos, Pinto, Praça et al. 2016b): é uma ferramenta cujo principal objetivo é a simulação de cenários com o propósito de estudar o comportamento dos MEE e os seus participantes. Para isto, o simulador tem incorporadas as ferramentas e características de três mercados europeus: MIBEL[3] (Península Ibérica), EPEX[4] (Europa Central) e NordPool[5] (Países Nórdicos). Adicionalmente, o simulador permite a participação tanto no mercado em bolsa, como no de ajustes, que variam nas suas restrições e perspetivas temporais. Por outro lado, também é possível modelar as características, interesses e comportamento das entidades que constituem o mercado (consumidor, produtor, agregadores, operador de mercado e operador de sistema). De forma a representar adequadamente a complexidade e dinamismo da interação entre as entidades participantes, o simulador é desenhado sob uma arquitetura multi-agente, que usa a plataforma JADE.

[3]MIBEL - `http://www.mibel.com/`
[4]EPEX - `http://www.epexspot.com/en/`
[5]NordPool - `https://www.nordpoolgroup.com/`

- MASGriP (Oliveira et al. 2012): é um simulador multi-agente que modela a operação interna das REI. Para tal, considera as entidades envolvidas através de agentes. De forma a validar a qualidade da gestão da rede feita pelo simulador, este permite a interligação com o simulador MASCEM de forma a verificar os benefícios da utilização da simulação do gestor de rede, através da análise dos lucros das entidades. O MASGriP modela: a rede de distribuição; consumidores domésticos, comerciais, industriais e rurais, que consideram micro-geração, veículos elétricos e DR; geração distribuída; parques de veículos elétricos; entidades agregadoras que permitem a participação de consumidores de pequena dimensão no MEE e eventos de DR; e operador da rede, responsável pela gestão de toda a rede. Este sistema é implementado recorrendo à plataforma JADE.
- *Simulator for Electric Power Industry Agents* (SEPIA) (Harp et al. 2000): é um simulador multi-agente que permite modelar operações físicas e de negócio dos SEE. Isto significa que possui componentes tais como geradores, baterias, entre outros, que estão conectados entre si através de nós da rede, fluxo monetário, etc (Mets, Ojea e Develder 2014). Nesta ferramenta é possível identificar três componentes: (i) interface gráfica que permite ao utilizador desenhar e monitorizar as simulações; (ii) agentes, que podem ser geradores, baterias e linhas; e (iii) simulação, que faz o acompanhamento do tempo de execução, gestão da troca de mensagens entre os agentes e aplicação de restrições de modelo. Este simulador tem na sua base um algoritmo de Q-learning que permite que os agentes aprendam com as suas ações e ajustem os seus comportamentos.
- *Smart Grids Information and Communication* (SGiC) (Haan et al. 2011b): é um simulador multi-agente que opera na *web* com o objetivo de fornecer suporte à decisão e análise de performance. As suas áreas de atuação são balanço energético, *virtual power plants*, e transporte de energia. Este simulador possui uma interface gráfica que permite a socialização dos seus utilizadores, considerando por exemplo programas DR, programas de gestão local, *virtual power plants* e gestão da demanda. Possui dados sobre operadores de rede, mercados, participação DR, entre outros.

No entanto, também existem outras ferramentas que estudam os SEE sob várias perspetivas, não correspondendo necessariamente a uma arquitetura multi-agente. Alguns exemplos são: DigSilent (DIgSILENT 2018); EuroStag (Antoine e Stubbe 1992); GridLAB-D (PNNL 2018); GridSim (D. Anderson et al. 2012); GridSpice (K. Anderson e Narayan 2011); *Hybrid Optimization of Multiple Energy Resources* (HOMER) (Homer 2018); OpenDSS (OpenDSS 2018); VpNET (W. Li et al. 2011); entre outros.

2.6.2 Co-Simulação e Simulação Integrada

Apesar da existência de diversas ferramentas para simulação e estudo dos SEE, grande parte delas são de domínios específicos, ou seja, atuam sob um determinado ponto de vista. Assim, interoperabilidade entre estes sistemas traz grandes benefícios, uma vez que permite ultrapassar as fronteiras de cada subsistema, e aproximar o estudo de um ambiente mais realista, onde estes domínios aparecem de forma integrada e correlacionada (Bastian et al. 2011).

Neste contexto, Mets, Ojea e Develder 2014 apresentam dois conceitos relacionados com a interoperabilidade entre sistemas: co-simulação e simulação integrada:

- Co-simulação: baseia-se na utilização de múltiplos simuladores, onde cada sistema possui a sua própria interface para a configuração necessária da ferramenta, controlo, e visualização de resultados (ver Figura 2.8). Deste modo, existe uma dificuldade acrescida na sincronização das ferramentas, uma vez que cada ferramenta pode possuir a sua própria noção temporal. Grande parte dos sistemas mantém as suas características quase intactas, não havendo um grande trabalho acrescido para a interoperabilidade, reduzindo o tempo de implementação e risco de erros. Contudo, esta abordagem pode trazer algumas desvantagens, nomeadamente no desperdício de tempo de simulação, especialmente em casos de execução sequencial. Além disto, também é introduzido tempo para pré-processamento entre a troca de comunicações, nomeadamente no mapeamento de dados de saída de um sistema *A* e dados de entrada de um sistema *B*.

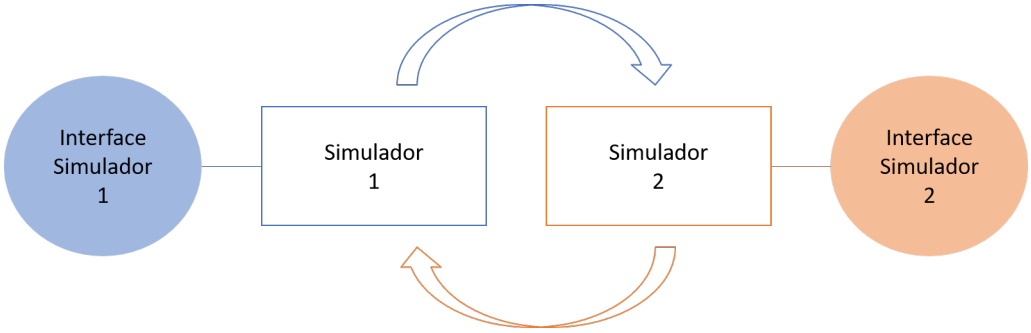

Figura 2.8: Co-simulação (adaptado de Mets, Ojea e Develder 2014)

- Simulação integrada: nesta abordagem, a simulação dos sistema acontece num único ambiente (ver Figura 2.9). Assim, existe uma gestão centralizada do tempo, configurações, dados, etc, permitindo efetuar simulações sem penalizações de tempo tão significativas. Todavia, é necessário a combinação dos modelos individuais de cada simulador constituinte, num modelo único, necessitando da demanda de mais esforço em implementação.

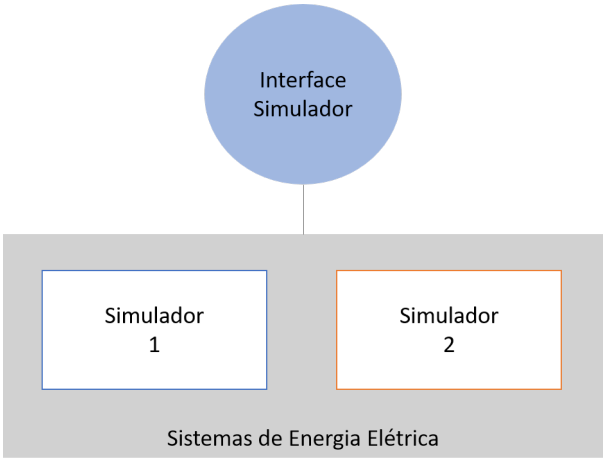

Figura 2.9: Simulação integrada (adaptado de Mets, Ojea e Develder 2014)

Padronizar a interoperabilidade entre vários sistemas é uma tarefa com grande dificuldade e complexidade, pois envolve a adaptação dos vários subsistemas de modo a que estes façam

uso dos padrões. Steinbrink et al. 2017 identifica a seguinte lista de requisitos necessários para facilitar a co-simulação ou simulação integrada em ambiente de REI, que podem ser aplicados aos SEE:

- Geração automática de cenários / formalização de cenários: é necessário a geração de conjuntos de cenários de forma automática, com uma descrição de alto nível.
- Interface de simulação / descrição de padrões: deve ser especificada uma descrição semântica do simulador, o que permite uma validação automática do cenário.
- Domínios adicionais de investigação: deve ser feita a investigação científica da integração com outros domínios, nomeadamente nas comunicações, fatores económicos, fatores sociais, problemas de segurança e fatores ambientais.
- Performance da otimização: a simulação deve ser distribuída em diversas máquinas.
- Gestão de dados: deve ser feita a gestão dos resultados durante a simulação, que permitam a sua análise, agregação e monitorização. Os dados devem ser armazenados adequadamente de forma a permitir a sua mineração.
- Visualização: ferramentas que permitam a monitorização e demonstração da simulação dos cenários.
- Quantificação de incerteza: o efeito de acoplar diferentes simuladores necessita de um estudo dos seus fatores de incerteza (diferentes resoluções, agregações do output, troca de dados, etc).

Tabela 2.7: Camadas genéricas para a interoperabilidade (Nguyen et al. 2017)

Camada	**Descrição**	**Problemas Associados**
Conceptual	Nível mais alto, onde os modelos são considerados como caixas negras (*black boxes*). Representação da interoperabilidade das ferramentas	Estrutura genérica da *framework*; Meta-modelos dos componentes
Semântica	Nível relativo à definição de papéis e conteúdo, em relação ao domínio	Especificação dos modelos individuais e das interações
Sintática	Nível que aborda a formalização da interoperabilidade	Formalização dos modelos individuais nos respetivos domínios; Especificação e manuseamento das diferenças entre os formalismos
Dinâmica	Nível relativo à execução da *framework*, envolvendo técnicas de sincronização e harmonização entre os diferentes modelos computacionais	Ordem de execução e causalidade dos modelos; Harmonização de diferentes modelos computacionais; Resolução de potenciais conflitos
Técnica	Nível relativo à implementação dos detalhes e avaliação da simulação	Implementação distribuída e centralizada; Robustez; Confiabilidade e eficiência da simulação

Nguyen et al. 2017 apresenta um modelo genérico estruturado em camadas para permitir o acoplamento dos sistemas, baseados nos modelos existentes para interoperabilidade genérica (Tolk e Muguira 2003) e multi-modelação (Siebert 2011). A Tabela 2.7 apresenta as camadas que constituem o modelo.

Portanto, por observação à Tabela 2.7, é possível encontrar cinco camadas: conceptual, semântica, sintática, dinâmica e técnica. A camada conceptual é relativa à arquitetura, onde é definido o processo de meta-modelação e topologia. A camada envolve três componentes: configuração do sistema de acordo com o cenário, propósito de investigação e definição do caso de estudo. Para além disto deve ser definida a estrutura semântica da integração dos sistemas. A camada semântica é a representação formal do sistema, envolvendo a definição dos papéis de cada subsistema e das suas relações, de forma a manter a compreensão entre subsistemas. É necessário que este modelo represente as diferentes escalas temporais e espaciais. É de ter em atenção que a representação semântica neste contexto refere-se ás relações entre sistemas e não dos conceitos a nível interno dos modelos. A camada sintática permite a reutilização dos modelos para diferentes cenários, pela harmonização dos modelos através de ontologias. A camada dinâmica baseia-se no processo de simulação. Define o modelo das interações/ações entre os subsistemas. A camada técnica refere-se à implementação e avaliação do sistema e dos seus modelos, considerando a robustez, confiabilidade e eficiência.

2.6.3 Ferramentas de Co-Simulação e Simulação Integrada

É possível encontrar na literatura ferramentas que permitem a co-simulação e/ou simulação integrada entre sistemas heterogéneos. Nesta dissertação serão abordadas as ferramentas EPOCHS, GECO e Mosaik.

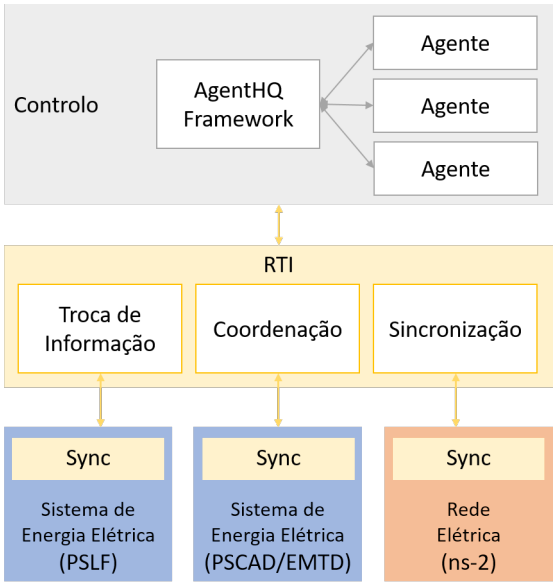

Figura 2.10: Arquitetura da ferramenta EPOCHS (adaptado de Mets, Ojea e Develder 2014)

O EPOCHS (Hopkinson et al. 2006) é uma plataforma de simulação multi-agente em ambiente distribuído, que tem como principal objetivo a integração de ferramentas de simulação

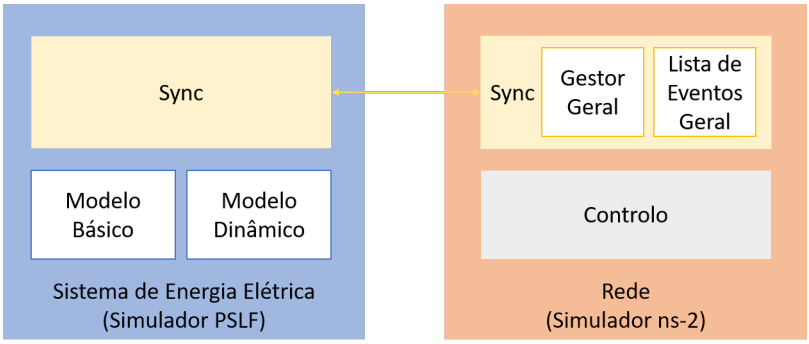

Figura 2.11: Arquitetura da ferramenta GECO (adaptado de Mets, Ojea e Develder 2014)

de componentes dos SEE e redes de comunicações. Assim, o simulador permite validar os benefícios e desvantagens da utilização de agentes para a proteção da rede de comunicação e monitorizar o sistema de forma a evitar "apagões"causados por colapso da tensão elétrica. Com vista à interoperabilidade entre os sistemas, esta *framework* serve-se de técnicas não intrusivas para interligar os sistemas, nomeadamente através de uma API. Na Figura 2.10 é possível visualizar a arquitetura da ferramenta e o modo como os componentes estão interligados. Numa perspetiva geral, o *Runtime Infrastructure* (RTI) desempenha o papel de interface entre todos os componentes do simulador, assegurando a sincronização das mensagens. O EPOCHS permite a interoperabilidade entre três simuladores: (i) *Power Systems Computer Aided Design* (PSCAD)/*Electromagnetic Transients including* (EMTDC) (PQ Soft 2018), que aliados formam uma ferramenta gráfica que permite simular respostas transitórias e resolução de equações diferenciais no âmbito dos sistemas eletromecânicos e eletromagnéticos; (ii) PSLF (General Electric 2018), utilizado para a modelação de estabilidade eletromecânica na rede elétrica; e (iii) *Network Simulator 2* (NS2) (University of Southern California 2018) para a simulação realista da rede. A infraestrutura é gerida pela entidade RTI, que tem como responsabilidade o reencaminhamento de todas as mensagens entre os componentes (simuladores), recorrendo a uma abordagem de sincronização por escalonamento do tempo. Este sistema possui o agente AgentHQ fornece uma visão unificada do modelo de simulação que é implementada pelos agentes inteligentes para, por exemplo, o EPOCHS simular esquemas de controlo e proteção distribuídos.

O sistema GECO (Lin, Sambamoorthy et al. 2011; Lin, Veda et al. 2012) é uma ferramenta para a simulação de redes e comunicação, e possui muitas características em comum com o EPOCHS. Esta ferramenta é usada para avaliar esquemas de monitorização, proteção e controlo da rede numa grande área (Mets, Ojea e Develder 2014) através da utilização dos simuladores NS2 e PSLF. A arquitetura do GECO está ilustrada na Figura 2.11. A ferramenta utiliza o simulador NS2 para a gestão global dos eventos armazenados numa lista para o efeito. A comunicação entre o NS2 e o PSLF é feita através de protocolos de comunicação baseados em *Transmission Control Protocol* (TCP) e *User Datagram Protocol* (UDP), e referem-se à troca de informações como dados dos SEE e comandos de controlo. Possui dois esquemas de proteção de rede, baseados em supervisão e *Ad hoc*, permitindo uma proteção da rede de backup mais rápida do que os esquemas tradicionais.

A ferramenta Mosaik (Scherfk 2018; Schütte 2013) é uma estrutura modular que permite a simulação de REI através da co-simulação de sistemas heterogéneos de simulação. A *framework* dispõe de uma API que permite efetuar simulações que envolvem vários simuladores

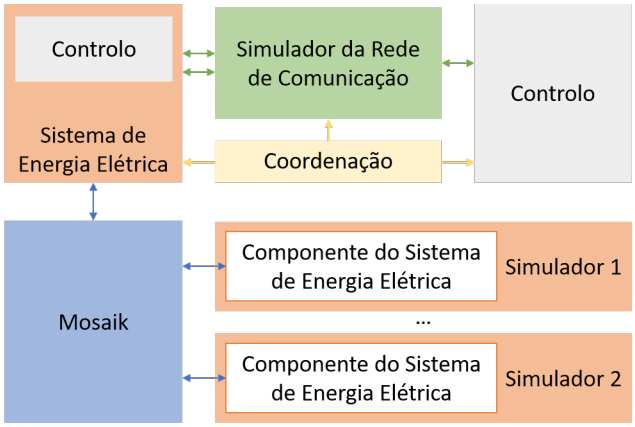

Figura 2.12: Comunicação abstrata entre a *framework* de co-simulação e o Mosaik (adaptado de Schütte 2013)

e troca de dados entre eles. Na Figura 2.12 está ilustrado o modelo abstrato de comunicação entre o Mosaik e as *frameworks* de co-simulação. O Mosaik possui dois componentes: interface de simulação (SIM API), que deve ser implementada pelos modelos de simulação a integrar com a *framework*; e o *Master Control Program* (MCP), que gere a composição dos cenários de simulação e controla a sua execução. Para este efeito, é adotada uma arquitetura baseada em camadas que dão suporte à composição e simulação dos diferentes cenários. São identificadas quatro camadas principais: sintaxe, semântica, cenário e controlo. O nível sintático especifica as interações entre os modelos de simulação. Para a integração de um novo subsistema é necessário que o utilizador implemente uma interface pré-definida (SimAPI e RPC API), de forma a garantir a gestão temporal da simulação e configuração dos modelos. O nível semântico adiciona uma descrição semântica ao modelo de dados, isto é, que dados podem ser trocados ao usar as interfaces disponibilizadas. O nível relativo ao cenário é uma descrição formal da simulação a realizar e depende do metamodelo. Por último, a cama de controlo fornece uma API com estratégias de controlo, para analisar e manipular o sistema em execução.

A Tabela 2.8 apresenta uma perspetiva geral das características destas ferramentas, como resultado da análise de Lin, Veda et al. 2012; Mets, Ojea e Develder 2014; Schütte 2013. Os requisitos analisados resultam dos tópicos considerados na secção 2.6.3, bem como dos objetivos desta dissertação. Segue-se uma breve descrição dos requisitos:

1. Sistema aberto à integração de novos subsistemas: permitir a interoperabilidade com novos sistemas, além dos que foram mencionados.
2. Facilidade na integração de novos sistemas: esforço necessário por parte do utilizador para integrar um novo subsistema, para que este possa ser usado na composição de cenários conjuntamente com as outras ferramentas já existentes.
3. Escalabilidade do cenário: permitir lidar com cenários de grandes dimensões.
4. Flexibilidade da composição do cenário: a definição de cenários deve permitir flexibilidade na construção de cenários, nomeadamente através das combinações possíveis entre os subsistemas, utilização de modelos, personalização de modelos, utilização de dados históricos, entre outros.

5. Ferramentas de controlo: ferramentas que permitam o acompanhamento da simulação, a sua gestão e análise de resultados.
6. Disponibilidade para utilização: se é ferramenta *open-access*.
7. Abrangência de domínio: abordagem dos diversos subsistemas dos SEE.
8. Utilização de modelo semântico: o modelo semântico permite a partilha de vocabulário entre os vários sistemas, independência dos modelos da componente de programação, validação automática dos modelos, entre outros.
9. Simulação em tempo real: se as ferramentas suportam a simulação de cenários em tempo real.
10. Simulação distribuída: distribuição dos agentes por diversas máquinas de forma a agilizar o processamento da simulação.

No preenchimento da tabela é considerada a seguinte nomenclatura:

- "*?*" sem informação suficiente;
- "-" não satisfaz o requisito;
- "*+*" satisfaz o requisito;
- "*+/-*" satisfaz o requisito parcialmente.

Tabela 2.8: Sumário da avaliação das ferramentas de co-simulação

Requisitos	**GECO**	**EPOCHS**	**Mosaik**
1. Sistema aberto à integração de novos subsistemas	?	?	+
2. Facilidade na integração de novos sistemas	?	?	+/-
3. Escalabilidade do cenário	+	+	+/-
4. Flexibilidade da composição do cenário	-	-	+/-
5. Ferramentas de controlo	+	+	+
6. Abrangência de domínio	-	-	+/-
7. Disponibilidade para utilização	?	?	+
8. Utilização de modelo semântico	-	-	+
9. Simulação em tempo real	-	-	+
10. Simulação distribuída	-	+	-

Numa perspetiva geral, é possível verificar que o sistema Mosaik é aquele que mais satisfaz os requisitos especificados. Contudo, a definição de cenários não possui a flexibilidade e escalabilidade desejadas para a solução do problema identificado nesta dissertação.

2.7 Considerações Finais

A utilização de fontes de energia renováveis é uma das principais preocupações da sociedade atual. A UE vem a modificar a sua legislação para implementar políticas que promovem o uso de energia renovável e desencorajam a emissão de gases de estufa, contribuindo para o aumento da pegada ecológica. A mudança dos moldes em que o setor energético opera

faz com que seja necessário adaptar o comportamento dos utilizadores da rede, bem como desenhar novos modelos de funcionamento, que abordam todos os segmentos

Existem diversos simuladores que procuram estudar o comportamento dos SEE de modo a prever o futuro do setor e antecipar o desenho de modelos e estratégias que o tornem mais sustentável, fiável e eficiente. No entanto, sendo este um setor com um elevado grau de complexidade, analisar o problema sob pontos de vista particulares não permite uma análise completa das necessidades do setor.

Desta necessidade é importante que sejam desenhadas soluções que permitam a reutilização de ferramentas que possibilitam o estudo de problemas específicos, combinando as suas capacidades individuais numa solução que permita analisar problemas com uma dimensão mais abrangente, e que também são capazes de lidar com a complexidade e escabilidade inerente à área.

Os SMA são soluções amplamente utilizadas para o desenvolvimento de soluções para estudo dos SEE, pois disponibilizam um conjunto de características que permitem uma boa representação do ambiente real e capacidade de processamento. Existem muitos simuladores com base em SMA, para o estudo de diferentes áreas: redes energéticas, mercados, apoio à decisão, distribuição, entre outros.

Sendo que para estas ferramentas possam agregar as capacidades de ferramentas com características heterogéneas, é necessário que estas consigam comunicar entre si. A utilização de semântica nas comunicações permite não só a troca de informação mas também de conhecimento, onde os dados são acompanhados do significado dos seus conceitos e relações.

Existem algumas soluções que já procuram realizar este trabalho, nomeadamente o GECO, o EPOCHS e o Mosaik, que efetuam a conversão de dados entre os sistemas. Contudo, estas ferramentas revelam limitações, pois ou permitem a interligação de sistemas do mesmo domínio de aplicação, ou então falham em permitir a criação de cenários de forma dinâmica e flexível.

A flexibilidade na criação de cenários revela-se importante, pois o utilizador, para efetuar as simulações pretendidas, necessitará de selecionar as ferramentas que podem ser interligadas, a sequencia com que elas executam, e introduzir os dados e modelos que são necessários preencher para que estas funcionem corretamente. A diversidade de informação que o utilizador precisa de definir para simular um cenário é uma tarefa morosa e suscetível a erros de introdução por mão humana.

O Capítulo 3 descreve a solução proposta no âmbito desta dissertação que pretende colmatar as falhas encontradas nas atuais soluções para a interoperabilidade, em que, além de permitir interoperabilidade entre ferramenta heterogéneas, também permite uma completa abrangência a outros domínios, e também disponibilizar um mecanismo simples para a construção de cenários.

Capítulo 3

Design e Desenvolvimento da Solução

3.1 Introdução

De modo a acompanhar a evolução dos SEE existem diversas ferramentas de estudo dos seus diferentes segmentos que permitem a compreensão do comportamento atual do setor, mas também das suas necessidades futuras. Contudo, sendo esta uma área com um elevado grau de complexidade, estudar os diferentes segmentos de forma individual não permite uma análise sensível ao impacto da interação das várias ramificações entre si, e consequentemente a obtenção de resultados mais realistas e confiáveis.

O GECAD é uma unidade de I&D que vem a desenvolver várias ferramentas de estudo e simulação dos SEE, e que tem investido em soluções que permitam a sua interoperabilidade, fazendo com que estas se comportem como uma só ferramenta capaz de abranger diversos domínios dos SEE. Com este objetivo, estão a ser desenvolvidas ontologias de domínio que permitem modelar os diferentes conceitos dos SEE e assim recorrer à sua utilização nas comunicações entre os sistemas. Entre elas, estão modelados conceitos relacionados com REI, MEE, veículos elétricos, entre outros.

Contudo, a configuração e execução de cenários tão abrangentes acarretam vários desafios. Quando é desejado estabelecer interoperabilidade entre duas ou mais ferramentas heterogéneas existem uma série de passos que consumem uma elevada quantidade de tempo e atenção do utilizador (ex.: configurar cada ferramenta de forma manual, efetuar a sua execução, analisar os resultados obtidos, mapear resultados para o modelo de entrada da outra ferramenta, entre outros), e por isso são muito suscetíveis a erro humano. Assim, quando é necessário aumentar a dimensão do cenário, o grau de complexidade inerente à sua criação também aumenta significativamente. Portanto, é essencial encontrar mecanismos que permitam simplificar este processo, onde além de diminuir o tempo e esforço necessários na fase de configuração, também aumenta a eficiência durante a execução.

A ferramenta *Tools Control Center* (TOOCC) foi criada no âmbito desta dissertação com o propósito de permitir a construção e execução de cenários que incluem vários domínios dos SEE, e que necessitam de recorrer a diferentes ferramentas para chegar à solução pretendida. Para tal, a TOOCC age como facilitador na interoperabilidade entre sistemas heterogéneos desenvolvidos pelo GECAD através do uso de ontologias. Este sistema foi implementado de forma modular. Esta ferramenta é apresentada na secção 3.2, onde são enunciados os requisitos que a ferramenta deve cumprir e as suas funcionalidades.

O presente capítulo apresenta o conceito e desenvolvimento da solução proposta, e encontra-se subdividido em oito secções principais, que permitem o acompanhamento do processo de desenvolvimento da solução, desde a formação do seu conceito até à implementação.

Após esta secção introdutória, a Secção *Tools Control Center* (3.2) tem como objetivo apresentar o conceito da solução pretendida no âmbito desta dissertação e definição da sua estrutura. Com esta finalidade são apresentados os requisitos do sistema, as funcionalidades identificadas para o utilizador, alguns pressupostos a considerar durante o desenvolvimento do projeto e, por fim, a arquitetura adotada para a implementação da ferramenta.

As secções seguintes descrevem os diferentes módulos identificados na arquitetura, nomeadamente: o modelo multi-agente (Secção 3.3), onde são apresentados os agentes utilizados, a sua arquitetura, comportamentos e implementação; o modelo de dados (Secção 3.4), que descreve os repositórios desenvolvidos para a solução, bem como outros que são também são utilizados; o modelo semântico (Secção 3.5), que explica quais são as ontologias utilizadas e como são aplicadas neste contexto; e a interface gráfica do utilizador (Secção 3.6), que estabelece a ponte entre o utilizador e os módulos mencionados.

No final do capítulo, são apresentadas algumas considerações finais (Secção 3.7), onde é resumido o conteúdo abordado ao longo das diversas secções.

3.2 Tools Control Center

A ferramenta TOOCC é uma solução que pretende agir como facilitador entre ferramentas heterogéneas, permitindo com que estas sejam capazes de partilhar o mesmo vocabulários e conceitos, e assim comunicar entre si para a execução de cenários relativos aos SEE. Consequentemente, é possível efetuar a simulação de cenários mais complexos que resultam da junção das capacidades individuais de cada ferramenta incorporadas, fazendo com que sejam considerados mais domínios dos SEE no mesmo cenário, e obter resultados mais realistas e precisos.

Atendendo que a introdução de dados e configuração de cada ferramenta pode ser uma tarefa complexa e trabalhosa, é pretendido que a ferramenta TOOCC disponibilize uma etapa de configuração do cenário, onde é possível efetuar a importação de dados de entrada e a escolha dos parâmetros dos vários sistemas a interagir.

Durante a execução/simulação das ferramentas é necessário atender às suas necessidades. Cada uma possui o seu domínio, arquitetura, dependências e modelos, e portanto é necessário que a TOOCC sejam capaz de lidar com este tipo heterogeneidade. Por outro lado, a eficiência é um ponto importante e é crucial que seja possível efetuar a execução no menor tempo possível.

Por fim, também é pretendido que a TOOCC forneça apoio à decisão sob o ponto de vista das diferentes entidades dos SEE (ex.: operador de rede, consumidores, produtores, agregadores, etc), tendo em conta diferentes horizontes de tempo. Assim, é possível efetuar projeções futuras do comportamento de uma rede, ou efetuar a gestão energética de uma casa para a hora seguinte, respeitando determinadas restrições. Com esta finalidade, pretende-se que a TOOCC permita a comparação entre resultados, não só finais como também relativamente a fases intermédias de simulação.

Portanto, como se pode observar pela Figura 3.1, o sistema vai compreender essencialmente três fases: criação e configuração dos cenários, simulação dos mesmos e a análise dos resultados obtidos.

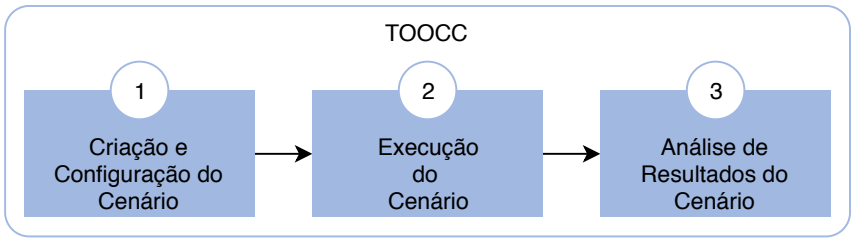

Figura 3.1: Processo geral para a simulação de cenários TOOCC

3.2.1 Requisitos do Sistema

Segue-se uma listagem detalhada com os principais requisitos que a ferramenta pretende cumprir:

- Interoperabilidade: união estratégica das capacidades individuais de ferramentas heterogéneas, permitindo expandir o domínio de conhecimento e simular problemas mais abrangentes. A relação entre as diversas componentes dos SEE permite uma melhoria na disponibilização dos resultados bem como a análise dos resultados;
- Flexibilidade: de modo a criar cenários com diferentes características é necessário flexibilidade na definição dos mesmos, onde os modelos que alimentam as ferramentas utilizadas podem ser personalizados, bem como a variação na combinação das ferramentas. Isto também significa que um cenário pode ser constituído por uma ou mais ferramentas, que podem ser executadas em simultâneo ou em paralelo consoante a necessidade do cenário em questão;
- Realismo: os modelos de entrada das diferentes ferramentas podem ser alimentados por dados reais já recolhidos e armazenados, de forma a conferir mais realismo na simulação do cenário. Estes dados podem ter diversas origem e podem estar armazenados em bases de dados ou ficheiros. Um exemplo prático seria a utilização de dados históricos de consumo de um consumidor doméstico, que irá alimentar um algoritmo de previsão;
- Gestão Centralizada: a ferramenta TOOCC deverá compreender todas as fases de simulação de um cenário, desde a sua criação à análise dos resultados. Esta é uma característica que pretende tornar o processo de configuração de toda a simulação mais fácil, sendo que este pode estar a lidar/configurar várias ferramentas em simultâneo e que, quanto mais ferramentas considerar, mais complexo se torna o processo;
- Análise de Resultados: esta solução deve disponibilizar os resultados obtidos pelas ferramentas, bem como estabelecer comparação entre os mesmos. Estes podem não ser apenas os resultados finais, mas também deve ser possível analisar resultados intermédios que podem ser úteis para uma melhor compreensão do problema. A forma como estes resultados são apresentados podem variar de formato consoante o seu tipo e objetivo;

- Automatismo: uma das maiores vantagens da ferramenta consiste na automatização da simulação que é resultante da configuração do cenário. Esta capacidade traz diversas vantagens, tais como a redução do erro humano durante a configuração das ferramentas e mapeamento de dados, aumento do desempenho, redução do tempo despendido na fase de construção do cenário e configuração das ferramentas, abstração de modelos, entre outros.
- Visualização Gráfica: muitas das ferramentas desenvolvidas pelo GECAD não possuem uma *Interface Gráfica do Utilizador* (GUI). Portanto, através da TOOCC é disponibilizada uma interface gráfica que permita a configuração de cada ferramenta, acompanhamento da sua execução e visualização dos resultados. Para além disso, é importante que a GUI seja capaz de permitir que todas as fases da configuração do cenário e simulação sejam intuitivas para o utilizador.

Além dos requisitos enumerados a cima, pretende-se que este sistema seja direcionado para pessoas conhecedoras do domínio, que são capazes de compreender os conceitos expostos. Muitas vezes estes utilizadores podem ser os criadores das ferramentas integradas, que tiram partido da TOOCC para interligar o seu trabalho com outras ferramentas e assim melhorar as suas contribuições científicas.

3.2.2 Funcionalidades do Sistema

Ao estabelecer interoperabilidade entre ferramentas de simulação de diferentes domínios dos SEE, são vários os conceitos necessários para modelar os diferentes cenários. Para além disso, algumas destas ferramentas também possuem parametrizações específicas que podem influenciar os resultados. A *framework* é desenhada para ser utilizada por um perito da área, que também necessita de ter conhecimento sobre como funcionam os sistemas a integrar. As Figuras 3.2 e 3.3 permitem identificar as principais funcionalidades do sistema. A Figura 3.2 apresenta o diagrama de casos de uso com as funcionalidades que o utilizador tem acesso ao utilizar a ferramenta. Por outro lado, a Figura-3.3 demonstra uma perspetiva das funcionalidades integradas numa perspectiva conceptual da ferramenta, sendo mais evidente a forma como estas se relacionam com os restantes elementos.

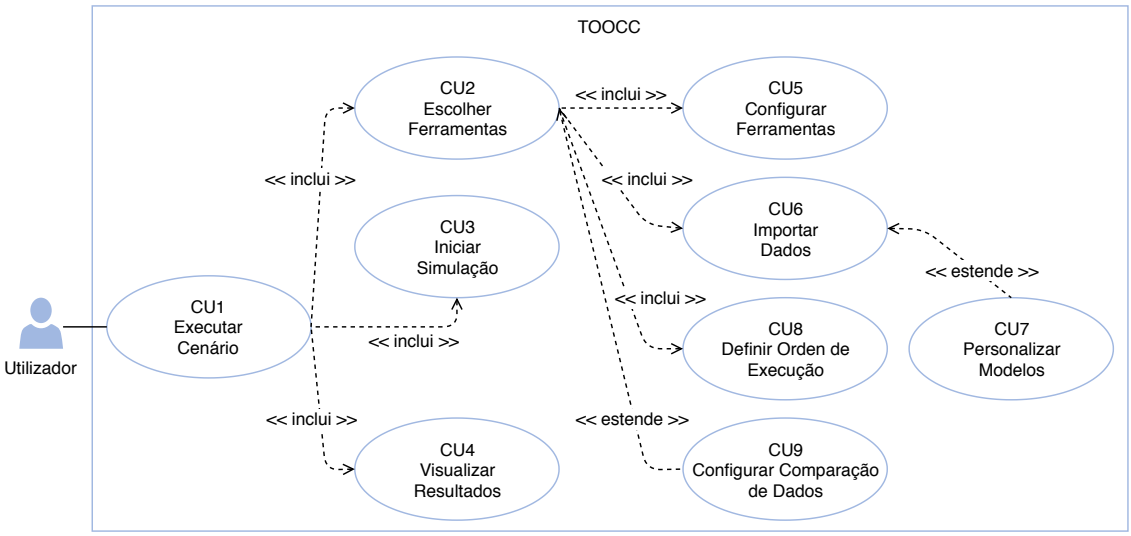

Figura 3.2: Diagrama *Unified Modeling Language* (UML) de Casos de Uso

O utilizador do sistema tem como função principal a execução do cenário (CU1). Para tal, o utilizador precisa realizar três ações: escolher que ferramentas vão compor a simulação (CU2), ordenar o inicio de simulação (CU3) e visualizar os resultados obtidos (CU4). Contudo, a escolha de ferramenta pressupõe outras tarefas, que podem ou não ser obrigatórias. Ao escolher uma ferramenta é necessário indicar os parâmetros de configuração das ferramentas escolhidas (CU5), e os dados de entrada, que são compostos pelos dados que vão preencher o modelo da área dos SEE (CU6). Possivelmente, existirão casos particulares que o utilizador não necessitará de efetuar estes dois passos, particularmente em situações em que a ferramenta não necessita de ter um modelo preenchido, ou que não existam parametrizações disponíveis. Quando são importados os modelos, é possível proceder à sua personalização (CU7), ou seja, ajustar/editar os seus valores. Quando existe mais do que uma ferramenta incluída no cenário, é necessário indicar ao sistema a ordem de execução das ferramentas (simultâneo ou sequencial) (CU8). E por último, existe a hipótese de configurar as comparações diretas entre resultados obtidos (CU9). De seguida, o sistema está pronto para iniciar a simulação e, por fim, apresentar os resultados.

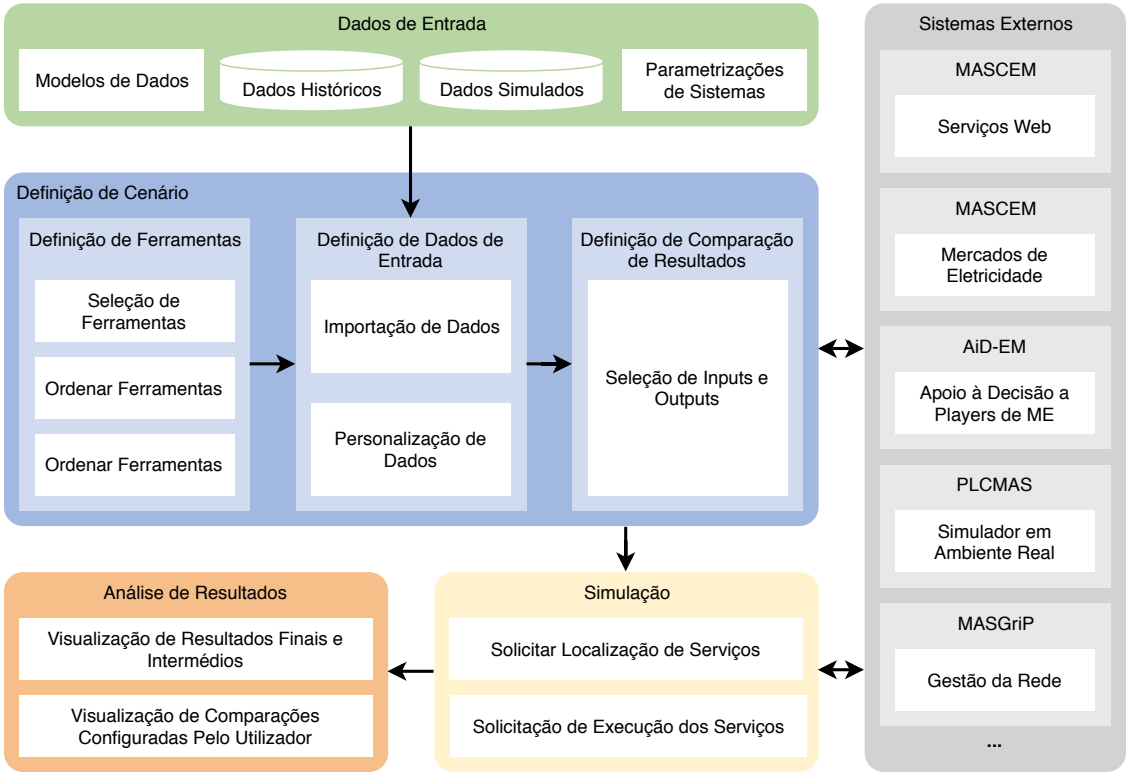

Figura 3.3: Perspectiva geral do conceito da ferramenta TOOCC

Definição de Ferramentas

Quanto um utilizador pretende dar início à criação de um cenário, o primeiro passo consiste na seleção das ferramentas que serão utilizadas na execução do cenário. No entanto, dependendo da ferramenta escolhida, a sua execução pode ser mais ou menos autónoma. Devido a este critério, esta dissertação considera que podem ser integrados dois tipos de sistemas: SBA e serviços. Os SBA são aqueles que possuem inteligência e independência na sua execução. Muitas vezes, estes até podem necessitar de interações com outros SMA

para executarem as suas tarefas. Já os serviços não são considerados sistemas inteligentes e executam tarefas menos complexas. Na maioria dos casos, estes serviços também serão executados por outras ferramentas que utilizam estes serviços para decompor a solução do problema em partes com menor dimensão, e estão disponíveis sob a forma de serviços *web*.

No âmbito desta dissertação, a ferramenta TOOCC integra outras soluções desenvolvidas pelo GECAD. Os SMA incluídos são os simuladores MASCEM, ALBidS, MASGriP e *Programmable Logic Controller Multi-Agent System* (PLCMAS). Os sistemas SMA MASCEM (simulação de MEE), ALBidS (fornecimento de apoio à decisão em MEE) e MASGriP (gestão da rede) já se encontram descritos na secção 2.6.1. O PLCMAS é uma ferramenta multi-agente que permite a simulação de cenários em ambiente real, sendo capaz de aplicar os resultados de outras ferramentas a dispositivos físicos, fazendo com que estes atuem de acordo com o pretendido. Os tipos de dispositivos disponíveis para estas simulações são luzes, tomadas de energia e dispositivos de ar condicionado. Para o SMA ser capaz de controlar os dispositivos, estes estão conectados a um *Programmable Logic Controller* (PLC). Para além dos SMA também estão disponíveis alguns serviços. Esses serviços são referentes a algoritmos de otimização da rede energética pela perspetiva do agregador, gestão energética em edifícios e previsões recorrendo a redes neuronais. De modo a estabelecer comunicação com estes serviços, o TOOCC recorre a um catálogo de serviços (Canito et al. 2019) que lista todos os serviços que ele pode utilizar, a localização, e definição dos modelos de entrada e saída que estes necessitam para a sua execução. Este catálogo apenas está disponível para uso interno do GECAD.

Quando as ferramentas estão escolhidas, é necessário indicar quando (em que instante de execução) as ferramentas vão iniciar a sua execução. Caso o cenário seja composto por SMA capazes de efetuar uma execução autónoma, o papel da ferramenta TOOCC prende-se em apenas dar início ao processo. No entanto, caso exista dependências entre as ferramentas escolhidas, o utilizador tem de indicar a ordem de execução e especificar como estas se interligam. Portanto, duas (ou mais) ferramentas podem executar em forma simultânea ou sequencial, dependendo da necessidade do cenário. Uma ferramenta A que dependa do resultado de uma ferramenta B (e que não têm inteligência para comunicar diretamente com a outra ferramenta), vai ter de aguardar que a ferramenta B termine a sua execução para iniciar a sua. Neste caso, para além de definir a ordem, também será necessário indicar a correspondência entre *outputs* e *inputs*, ou seja, definir que o resultado obtido pela ferramenta B vai servir de dado de entrada da ferramenta A. Esta correspondência é feita através do uso de ontologias, que permitem a definição de conceitos e estabelecer comparações.

Definição de Dados de Entrada

De modo a permitir que os serviços e SBA possam ser executados, é necessário fornecer os dados de entrada necessários. Os dados de entrada, para além dos parâmetros de configuração, podem também incluir modelos constituídos por dados simulados ou reais. Para este propósito, existem duas fontes possíveis: (i) uma base de dados onde os dados e modelos estão armazenados, ou (ii) através da leitura de ficheiros com uma estrutura específica. Relativamente aos dados reais, estes referem-se a dados de consumo (ar condicionado, luzes e tomadas), condições meteorológicas, produção de energia (solar e eólica), e energia transacionada relativa aos MEE.

Os modelos concebidos são o resultado de um estudo dos requisitos necessários para modelar REI e MEE, que já foram testados e utilizados diversas vezes para diferentes tipos de problemas. Estes incluem implementação em grande escala e em tempo real de gestão de recursos energéticos; componentes das REI (geradores, unidades de armazenamento, baterias e carros elétricos); tarifas DR que permitem uma gestão flexível de consumo energético; tarifas de consumo de energia relativas aos Estados Unidos da América, Brasil e vários países da UE; modelos de agregação de entidades; perfis de negociação; propostas de mercado; entre outros. A grande variedade de modelos permite à ferramenta, juntamente com os simuladores externos, permitir a especificação de cenários diversificados, nomeadamente no contexto, características e objetivos. Adicionalmente, também é possível efetuar estudos do impacto destes modelos num determinado contexto.

Numa outra perspetiva, dependendo dos sistemas envolvidos na simulação, a resolução temporal dos dados/modelos também podem ser diferentes. Por exemplo, a TOOCC também pode processar simulações em tempo real, tirando partido de aparelhos atuadores normalmente presentes em edifícios, em contexto de laboratório.

Simulação e Interoperabilidade

Durante a fase de simulação, o papel do TOOCC pode diferir conforme as ferramentas envolvidas no cenário. Caso as ferramentas consideradas sejam SBA capazes de executar de forma autónoma, o papel do TOOCC prende-se com alinhar o início da execução dessas ferramentas e fornecer-lhes a informação necessária (dados e modelos) para iniciar o processo. Quando o cenário é composto por serviços onde é necessário definir a interação das ferramentas entre si, o papel do TOOCC consiste em servir de facilitador, ou seja, a cada iteração da execução a TOOCC vai agir como facilitador na comunicação entre as ferramentas e mapear os dados, como está exemplificado na Figura 3.4.

Figura 3.4: Exemplo de comunicação entre ferramentas externas e a TOOCC

De forma a permitir a interoperabilidade entre os sistemas externos, é utilizada semântica nas comunicações, que permitem aos sistemas externos a partilha do mesmo vocabulário de modo a compreender os mesmos conceitos, e prevenir diferentes interpretações da informação.

Neste trabalho é possível identificar dois tipos de ontologias. O primeiro tipo são as ontologias de domínio, que são a base para a comunicação entre sistemas. Estas ontologias permitem a descrição de vocabulário que é partilhado entre os sistemas, desde medidas (ex.: *energy ontology*, *temperature ontology*, etc.), a conceitos mais complexos como flexibilidade de cargas durante um evento de DR (ex.: *load flexibility ontology*) e gestão energética (ex.: *scheduling, optimization and forecasting ontology*). O segundo tipo de ontologias é relativo à parte processual dos sistemas - ontologias de aplicação - e são utilizadas para a definição do funcionamento da TOOCC.

Análise e Comparação de Resultados

A componente de análise e comparação de resultados permite demonstrar os resultados obtidos após a execução do cenário. Estes resultados podem ser visualizados através de gráficos, tabelas e ficheiros de output desenhados especialmente para este propósito. Além disso, para visualizar os resultados também é possivel efetuar comparações entre eles. Estes mecanismos permitem que o utilizador desenhe uma nova visão para os resultados, que lhe permitem retirar conclusões de forma mais eficiente, e dando informação adicional para análise e apoio à decisão.

A comparação de resultados também é feita conseguida através de ontologias, uma vez que estas partilham os mesmos conceitos e vocabulário, é possível detetar automaticamente quais são os dados que podem ser comparados e disponibilizar essa informação ao utilizador. Assim o sistema dá rapidamente a sugestão dos dados mais indicados a serem comparados.

3.2.3 Pressupostos

Durante a execução deste trabalho devem ser considerados alguns pressupostos, de modo a garantir a consistência da solução proposta, presente na seguinte lista:

- No âmbito desta tese não são consideradas questões relativas a segurança informática, nomeadamente a segurança nas comunicações, sistemas gráficos, entre outros. Apenas serão considerados aspetos básicos como validação de dados de entrada, etc.
- Sendo que este é um sistema criado no âmbito do trabalho desenvolvido pelo GECAD, as escolhas tecnológicas e arquiteturais devem contemplar a cultura da organização para que seja possível manter o sistema no futuro.
- As ferramentas externas ao TOOCC são tratadas como *blackboxes*, mas estas têm de assegurar os requisitos mínimos para a interoperabilidade, ou seja, serem capazes de receber comunicações com o exterior e executar a partir delas.

3.2.4 Arquitetura

A necessidade de estabelecer interoperabilidade entre sistemas heterogéneos é um dos principais requisitos para o sucesso deste trabalho. Deste modo, é essencial que a arquitetura do sistema permita a comunicação com diversos tipos de sistemas com diferentes linguagens de programação, bem como suportar a escabilidade do número de ferramentas suportadas e capacidade de processamento. Para além disto, a escolha das tecnologias e linguagens de programação deve ser feita de modo a respeitar a cultura da organização, as necessidades do sistema, e da sua futura manutenção ou expansão.

Com o objetivo de otimizar e estruturar adequadamente o código, foram adotados alguns padrões de desenho (*Design Patterns*) na conceção do código. Os padrões de desenho estabelecem soluções standard para problemas comuns no desenvolvimento de *software*, e dão suporte à escabilidade das aplicações. Frequentemente, estes padrões são classificados em três áreas: (i) padrões de criação, que resultam no adiamento ou abstração de objetos (ex.: *Factory* e *Singleton*); (ii) padrões estruturais que definem como as classes são compostas em objetos estruturalmente mais complexos (ex.: *Adapter* e *Façade*); e (iii) padrões comportamentais que possuem a capacidade de atribuir responsabilidades a outros objetos

(ex.: *Observer* e *Strategy*). Na lista que se segue estão identificados os padrões de desenho utilizados no âmbito desta dissertação, cuja definição é adaptada de Gamma et al. 1993:

- *Factory*: existe uma classe *interface* (*Factory*) que possui um método abstrato que é implementado nas subclasses da *Factory*. Desta forma, a decisão de como as instâncias são criadas é delegada às subclasses. É vulgarmente associado a *Strategies* ou *Adapters*.
- *Singleton*: é permitida apenas a criação de uma instância da sua classe, que é acedida de forma global. Este padrão é utilizado vulgarmente para aceder a um log de dados.
- *Strategy*: permite implementar diferentes soluções para o mesmo método. Ou seja, existem alternativas de atuação para um problema, e através de um critério de decisão é selecionada uma das alternativas.

Além dos padrões de desenho, numa perspetiva macro do sistema também foram utilizados padrões de arquitetura de *software*, que possuem diferentes características e propósitos. A adoção de padrões arquiteturais tem impacto direto na qualidade do sistema, nomeadamente nos requisitos não funcionais como é exemplo o desempenho. A lista seguinte apresenta os padrões arquiteturais utilizados no desenvolvimento da ferramenta TOOCC:

- Arquitetura Modular: optar por uma arquitetura modular é vantajosa pois o seu cariz permite a expansão e alteração de cada um dos seus módulos de modo individual, bem como facilita a integração de novos módulos no futuro. Além disto, a estruturação do código torna-se mais legível e permite decompor o sistema em subsistemas, com menor grau de complexidade.
- Arquitetura Modelo-Visão-Controlador (ou)*Model-View-Controller* (MVC)): este estilo de arquitetura resulta na separação entre o modelo de dados (*Model*), a interface com o utilizador (*View*) e a camada com as regras de negócio (*Controller*). A sua maior vantagem reside na separação da camada de negócio em que, quando é necessário efetuar alterações ao sistema as outras camadas não são afetadas. Este facto permite mais reutilização de código e flexibilidade.
- Arquitetura Orientada a Serviços (ou *Service-Oriented Architecture* (SOA)): este estilo de arquitetura é ideal para sistemas distribuídos, onde componentes com modelo de negócio são disponibilizados através de serviços *web*. Esta arquitetura habilita a integração de componentes desenvolvidos com diferentes tecnologias e plataformas de forma simples, assim como a sua reutilização em várias aplicações.

A Figura 3.5 ilustra os principais módulos considerados no desenvolvimento da ferramenta TOOCC. Nas secções seguintes, estes módulos serão apresentados em maior detalhe, descrevendo o seu propósito e implementação.

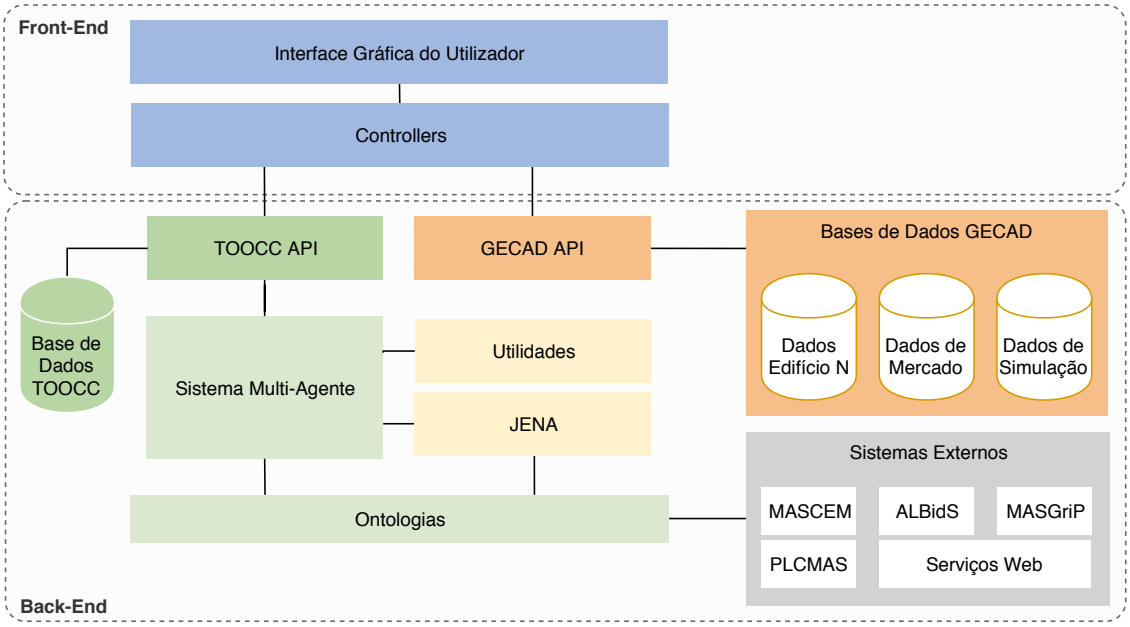

Figura 3.5: Arquitetura geral da ferramenta TOOCC

3.3 Modelo Multi-Agente

Um dos módulos que compõe a arquitetura da ferramenta TOOCC é o SMA. O SMA foi desenvolvido com a responsabilidade de executar a simulação dos cenários configurados, garantindo que a comunicação com os sistemas externos é estabelecida com a ordem definida pelo utilizador. A escolha tecnológica tem por base diferentes objetivos que a ferramenta pretende alcançar.

O primeiro requisito na escolha prende-se com a escabilidade da dimensão do cenário desenhado. Embora possam ser criados cenários com uma dimensão reduzida e que resultam em poucas comunicações, também podem ser criados cenários complexos que utilizam várias ferramentas e fases de execução. Nestes casos, é necessário que o problema seja decomposto em subtarefas que serão atribuídas a diferentes agentes. Assim, a execução é simplificada pela divisão do problema e pela definição de papeis, que na fase finalização da execução podem ser convertidos na solução desejada

Uma outra motivação trata-se da manutenção do sistema. Com o crescimento da *framework* TOOCC prevê-se o surgimento de novos tipos agentes com papeis ainda não existentes, como é exemplo a configuração de agentes que participarão na execução dos sistemas externos. Atendendo a esta necessidade, o tipo de arquitetura modular que um SMA oferece faz com que seja possível incrementar o número de agentes e comportamentos sem um elevado esforço na sua implementação, e evitando conflitos com a solução já existente.

A arquitetura distribuída dos SMA faz com que estes possam tirar partido das características de diferentes máquinas para o seu processamento. Desta forma, os agentes conseguem mover-se para máquinas do domínio preparadas para dar resposta a determinados tipos de problema, possuindo o sistema operativo, *software*, ou outro requisito necessário.

Considerando os pontos já referidos, no desenvolvimento deste trabalho optou-se pela *framework* JADE. Sendo um dos principais objetivos da TOOCC a interoperabilidade com

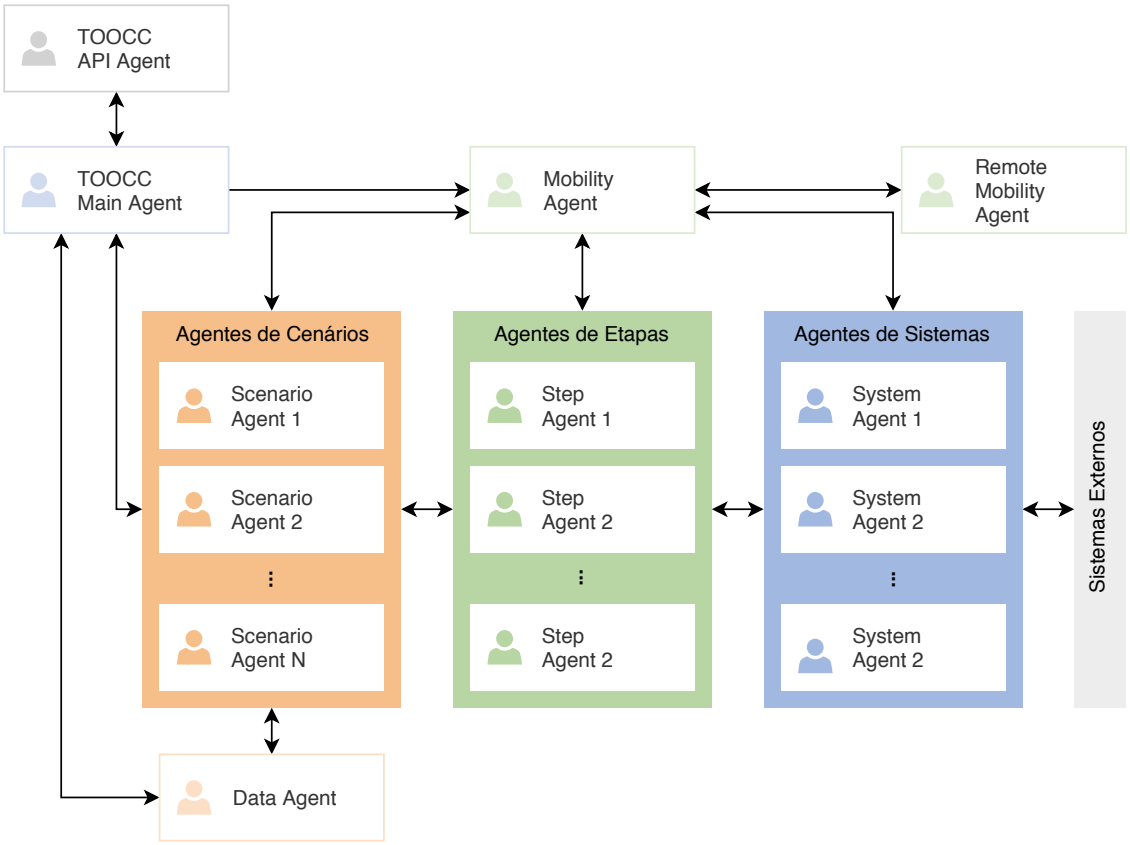

Figura 3.6: Modelo Multi-Agente

outros sistemas, é importante que a tecnologia escolhida seja de acordo com *standards* utilizados para o efeito. Assim, a FIPA promove a linguagem ACL como *standard* para a comunicação entre SBA. Além da troca de informação, também é possível agregar significado às mensagens trocadas através do uso de ontologias, que permitem a definição de sintaxe e semântica ao seu conteúdo. Consequentemente, os agentes tornam-se capazes de comunicar de forma mais completa, pois também possuem a componente de interpretação.

A Figura 3.6 apresenta o modelo multi-agente implementado em JADE. Este modelo tem como base seis tipos de agentes que proporcionam uma perspetiva conceptual da execução de um cenário, nomeadamente:

- TOOCC API Agent (ApiA): é o agente responsável por estabelecer a ponte entre a API TOOCC e o SMA. Este faz o pedido ao agente principal do SMA para efetuar a simulação, e fica a aguardar pelos resultados.
- TOOCC Main Agent (TMA): é o agente principal do SMA, responsável por iniciar e coordenar em alto nível toda a simulação. A partir dos dados fornecidos aquando da ordem de execução da simulação, este vai desencadear a criação de um Scenario Agent por cada cenário configurado. Após a execução de todos os Scenario Agents é este agente que dá por finalizada a simulação.
- Scenario Agent (ScenA): é responsável pela coordenação da execução de um cenário específico. Uma das suas funções reside em coordenar as fases de execução através da criação de Step Agents e garantir que o cenário é executado na sua totalidade.

- Step Agent (StepA): tem como função a gestão das comunicações com os sistemas externos, de forma paralela. Ou seja, vai proceder à criação de System Agents que irão comunicar em simultâneo com os seus respetivos sistemas externos, dentro da mesma fase de execução, permitindo agilidade na obtenção de resultados.
- Service Agent (ServA): vai estabelecer a comunicação direta com o sistema externo. Para isso deve ser conhecedor da semântica - ontologia - utilizada na comunicação. Assim que recebe o resultado pretendido notifica o seu criador (Step Agent).
- Mobility Agent (MobiA): tem como função decidir para qual máquina do domínio todos os agentes da simulação devem ser movidos. A decisão é tomada mediante um conjunto de critérios, nomeadamente sistema operativo e *software* instalado.
- Remote Mobility Agent (RMobA): está alojado na máquina do domínio pela qual o trabalho pode ser distribuído e estabelece comunicação com o MobiA, transmitindo-lhe a informação que ele precisa de saber para alojar o agente que se pretende mover.
- Data Agent (DataA): efetua a gestão centralizada dos dados da simulação, garantindo que os resultados são guardados corretamente, e que o estado de cada ponto da simulação é atualizado, garantindo a execução completa do sistema.

Para além dos agentes apresentados, também existem outros que fazem parte do SMA, mas que são nativos do JADE, como o Agente DF como diretório de páginas amarelas para a disponibilização de serviços, o Agente AMS para o controlo dos agentes e o Remote Agent Management (RMA) que fornece uma interface gráfica para a gestão e visualização do estado e comunicações dos agentes.

3.3.1 Processo de Criação de Agentes

Assim que é iniciado o processo de execução do cenário a primeira tarefa a realizar é a criação dos agentes necessários para a simulação. Só então é que o SMA está habilitado a passar à fase de execução do cenário. A Figura 3.7 apresenta o diagrama de sequência relativo às comunicações principais estabelecidas entre os agentes intervenientes.

Observando a Figura 3.7 é possível ver que o processo de criação dos agentes intervenientes inicia-se pelo ator do sistema (utilizador), quando dá ordem de início da simulação. Esta ordem é transmitida através da TOOCC API até ao agente principal do SMA - TMA - e assim são desencadeadas as ações seguintes. Primeiramente o TMA assegura-se que os agentes MobiA e DataA estão vivos no sistema. Embora estes agentes sejam usados pelos restantes, eles são independentes dos cenários. Ou seja, estes agentes são criados apenas uma vez e serão (re)utilizados para todas as simulações seguintes, até o TOOCC ser terminado.

De seguida são iniciados os agentes responsáveis por executar o cenário. Como já foi referido anteriormente, será criado um ScenA por cada cenário configurado para a simulação. Cada simulação irá executar em simultâneo e de forma independente. Estes cenários podem ter a mesma base e diferir em alguns pontos, ou possuir objetivos completamente distintos. O objetivo trata-se de permitir ao utilizador configurar vários cenários de uma só vez, poupando tempo no processo em geral. Imediatamente após a sua criação um ScenA vai criar todos os StepAs necessários e, assim que é recebida a notificação que todos os seus StepAs já estão prontos a executar, o ScenA informa o agente principal que também está pronto para iniciar a execução. Analogamente aos ScenAs, os StepAs possuem uma ordem de

execução especificada na fase de configuração, pois os resultados da fase anterior vão ser usados na fase seguinte. Dentro de cada fase de execução são considerados um ou mais sistemas externos. Portanto, cada StepA irá criar um ServA para comunicar com os sistemas externos, de forma paralela. Sempre que um agente é criado, no final do seu processo de criação ele notifica o seu criador que está pronto a executar, para que ele também possa aguardar pelos restantes agentes filhos e assim poder notificar o seu criador. Sempre que um ScenA notifica o TMA que está pronto, o agente principal vai verificar se já estão criados todos os agentes necessários para a simulação. Assim que esta condição estiver assegurada, o TOOCC está pronto para prosseguir com a execução da simulação.

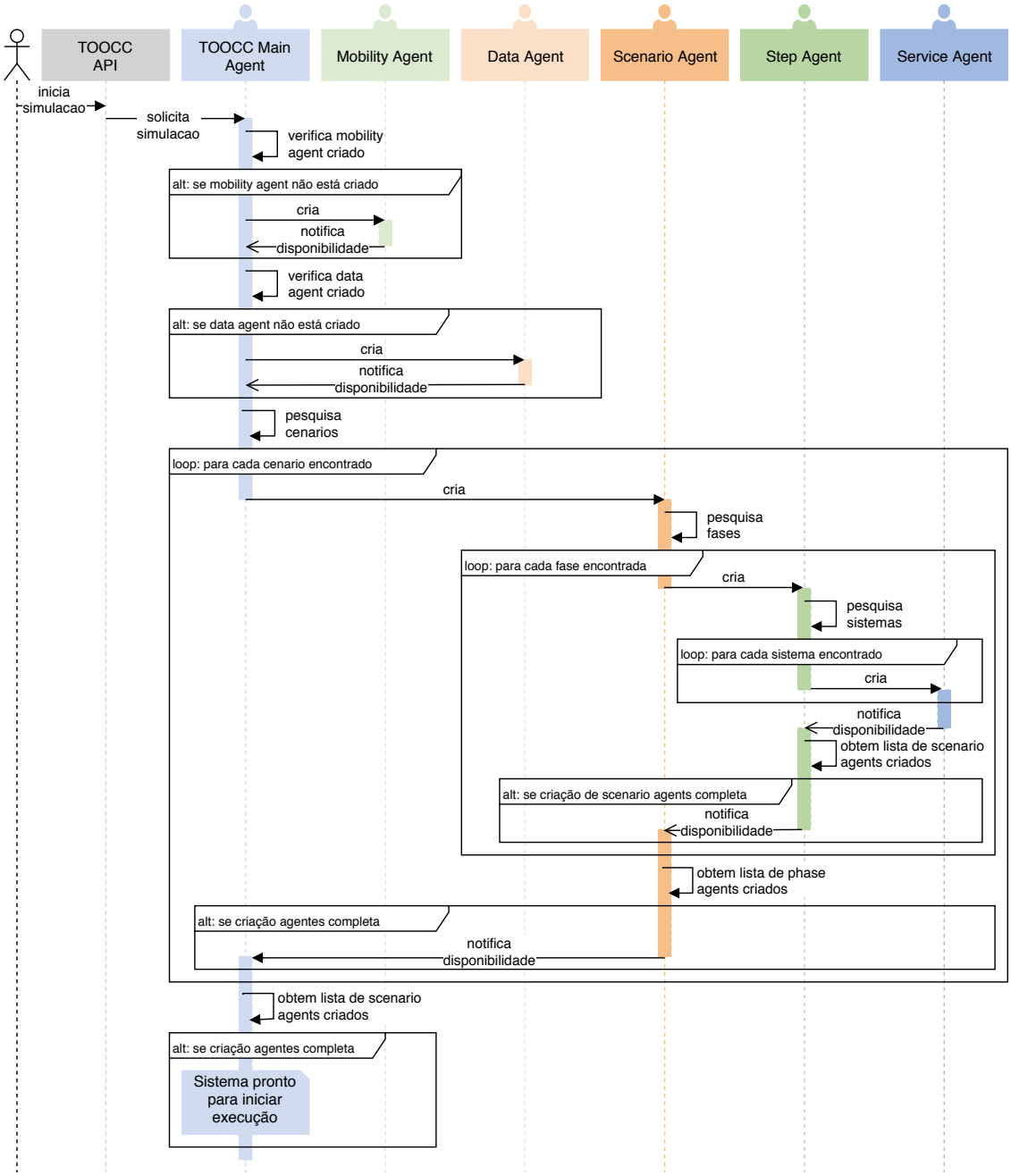

Figura 3.7: Diagrama UML de Sequência relativo à perspetiva geral das comunicações do sistema TOOCC para a criação dos agentes

Todos os agentes criados no âmbito deste trabalho respeitam uma arquitetura base, ilustrada na Figura 3.8. Uma das partes constituintes da arquitetura é a Base de Conhecimento (BC) (ou *Knowledge Base*). A BC é utilizada para guardar todas as informações necessárias para a execução do agentes, desde o seu modelo, dados necessários para a execução, endereço do agente que o criou (se aplicável), identificadores dos agentes subordinados (se aplicável) e outras informações úteis. Cada agente também possui Comportamentos (ou *Behaviours*) que vão ser adotados e que desencadeiam ações/processamento do agente. Estes Comportamentos podem iniciar juntamente com o agente ou provocados pela receção de uma mensagem de outro agente. Para isso, existe um Comportamento especial denominado `MessagesServerBehaviour` que é iniciado durante a criação do agente, e que age como recetor de todas as mensagens do agente e direciona para a execução do Comportamento adequado (ver secção 3.3.2).

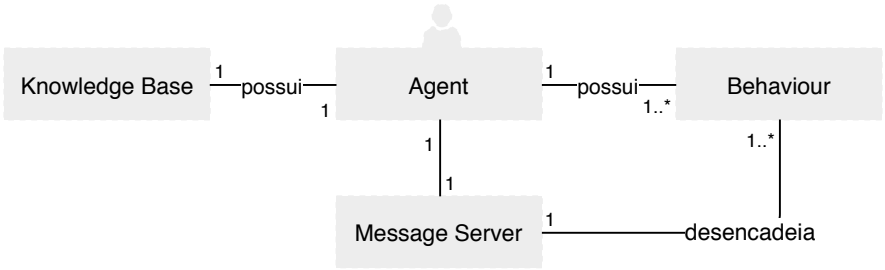

Figura 3.8: Arquitetura base de um agente

Durante o *setup* de criação de um agente são executadas diversas subtarefas. Além de iniciar o agente com os seus serviços, é efetuado registo do agente no DF, que possui o catálogo de todos os agentes e serviços disponíveis. Para além disso é iniciada a BC do agente e despoletados os comportamentos iniciais: o `MessagesServerBehaviour` e o `AskMachineBehaviour` (ver secção 3.3.3). Adicionalmente, os agentes que necessitam de criar agentes subordinados (como é o caso do ScenA e StepA), ao invés de considerar o comportamento `CreateScenarioAgentsBehaviour` que é desenhado para o TMA iniciar todos os seus agentes do tipo ScenA, este será substituído pelo respetivo, ou seja, `CreatePhaseAgentsBehaviour` para o ScenA criar os agentes tipo StepA, e `CreateSystemAgentsBehaviour` para o StepA criar agentes do tipo ServA. Por outro lado, o DataA e o MobiA não necessitam criar agentes subordinado, pelo que podem imediatamente notificar ao TMA que já se encontram prontos a executar, através da execução do comportamento `InformTOOCCMainAgentImAliveBehaviour`. Este comportamento só é executado pelos outros tipos de agentes quando o seu agente pai verifica que já estão todos os agentes criados (informação obtida através da sua notificação em como está pronto para prosseguir com a execução).

3.3.2 Receção e Interpretação de Mensagens

A receção de mensagens por parte dos agentes é efetuada através um Comportamento cíclico denominado por `MessagesServerBehaviour`, que executa continuamente à escuta de novas mensagens. A Figura 3.9 tem representado o fluxograma que exemplifica as ações executadas quando é recebida uma mensagem.

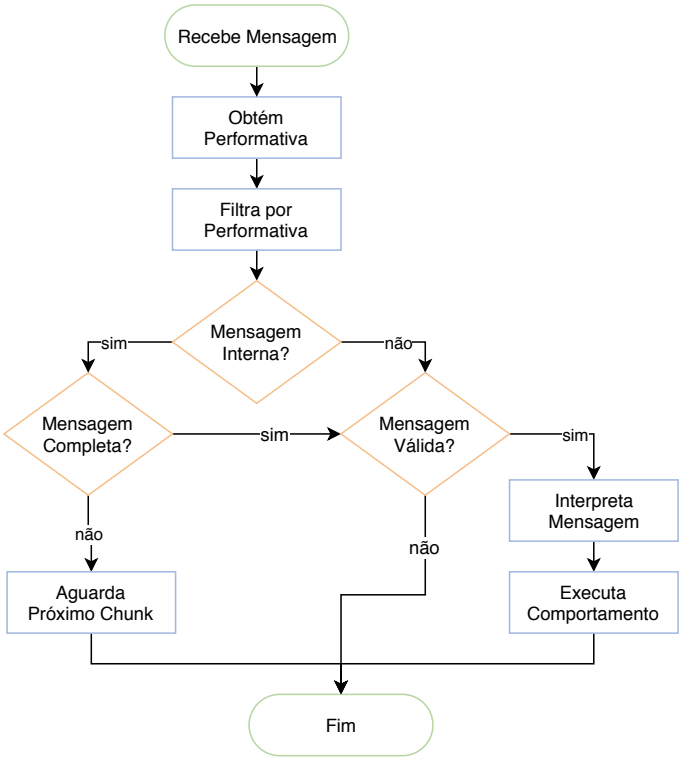

Figura 3.9: Fluxograma do comportamento MessageServerBehaviour implementado pelos agentes

Como é possível visualizar na Figura acima, primeiramente a mensagem recebida será filtrada consoante o tipo de performativa contido na mensagem ACL. Estes tipos são utilizados para enriquecer a mensagem transmitida, indicando o objetivo da mesma, e assim adotar o comportamento adequado. Existem diversas performativas disponíveis, nomeadamente: `AGREE`; `CANCEL`; `CONFIRM`; `INFORM`; `INFORM_IF`; `NOT_UNDERSTOOD`; `UNKNOWN`; entre outros. Quando é identificado o tipo de mensagem ACL recebido, será executado um método para interpretar as mensagens do tipo indicado, ou seja, caso seja uma performativa do tipo `REQUEST`, será executado o método `interpretRequest`. A divisão da interpretação por métodos ajuda na organização do código e facilita o entendimento do mesmo.

De seguida, caso o remetente da mensagem seja um agente interno do sistema, será verificado se a mensagem já se encontra pronta para ser interpretada. Durante toda a execução, a dimensão do conteúdo das mensagens trocadas pode tornar o processo mais lento. Desta forma, quando são realizadas comunicações a nível interno, os agentes vão segmenta-las em diversas partes (*chunks*). O recetor destas mensagens possui uma `MailBox` (Caixa de Correio) com a indicação de quantos segmentos foram construídos e apenas passa à fase de interpretação da mensagem assim que recebe todos os segmentos. Este procedimento aumenta severamente a eficiência, através da redução do tempo despendido na troca de mensagens. A Figura 3.10 ilustra o diagrama de classes relativo ao armazenamento de *chunks*.

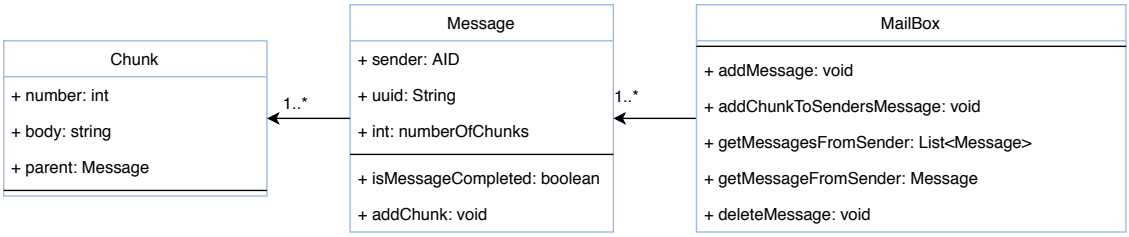

Figura 3.10: Diagrama UML de Classes (simplificado) relativo à caixa de entrada na receção de mensagens por parte dos agentes

Na interpretação da mensagem, se aplicável, será utilizada a ontologia indicada de forma a verificar se a mensagem está formada corretamente. Em caso afirmativo, procede-se então à execução do comportamento pretendido para o tratamento da mensagem recebida.

3.3.3 Distribuição de Agentes por Máquinas

Sendo uma das maiores vantagens da utilização dos SMA a sua arquitetura distribuída, aliando à necessidade de garantir a escabilidade do sistema, foi desenvolvido um agente que tem como função a distribuição dos agentes do cenário por diversas máquinas, como é possível observar na Figura 3.11. Para tal, o agente MobiA vai comunicar com um agente remoto - RMobA - instalado em cada máquina que pode ser utilizada na simulação.

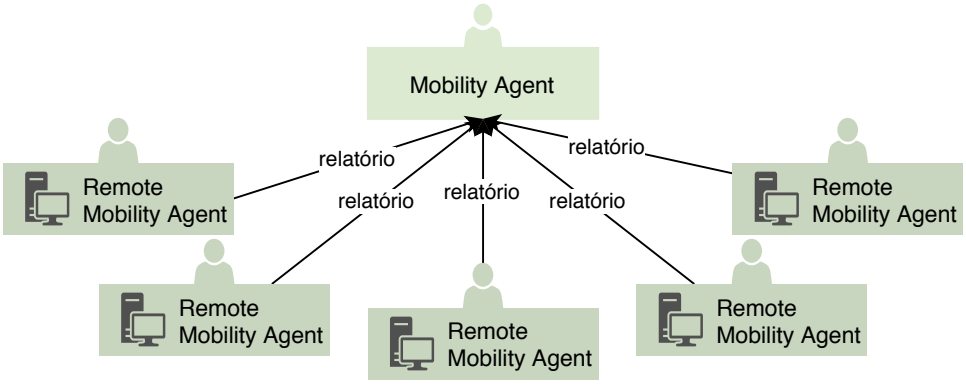

Figura 3.11: Diagrama representativo da distribuição dos RMobA pelas máquinas

O RMobA tem como objetivo a leitura das características da máquina onde se insere e enviar um relatório com as mesmas para o MobiA. Para este efeito o agente utiliza a biblioteca Java "snmp4j", que por via do protocolo Simple Network Management Protoco (SNMP), transmite as informações relevantes. O relatório define quais são as informações que devem ser recolhidas das máquinas. Para tal, é considerada a classe Report, que implementa uma interface Serializable, de forma a ser possível transportar a mensagem entre os contentores dos agentes. O relatório tem a seguinte informação:

- Endereço do contentor onde o agente está inserido;
- Sistema operativo da máquina;
- Arquitetura do sistema;

- Número de Processadores;
- Memória RAM disponível pela máquina;
- Bandwidth: largura de banda necessária;
- Nome das diferentes unidades de armazenamento;
- Espaço de memória disponível nas diversas unidades de armazenamento;
- Percentagens de utilização dos diferentes processadores;
- Largura de banda das diversas interfaces do sistema;
- Nomes das interfaces da máquina;
- Software instalado no sistema.

Numa outra perspetiva, os agentes a serem distribuídos precisam garantir que a máquina para onde serão transferidos possui as condições necessárias. Para isso, estes agentes precisam definir quais são os requisitos necessários na decisão da mobilidade, que estão presentes na sua base de conhecimento. A informação necessária para garantir a sua execução é constituída pelos seguintes pontos:

- Uuid: número identificador único de um agente;
- Name: nome dado para identificar um agente;
- Disk: espaço em disco necessário para a movimentação do agente;
- RAM: quantidade de memória RAM necessária;
- CPU: quantidade de processamento necessária;
- Bandwidth: largura de banda necessária;
- OS: sistema operativo em que o agente deve ser inserido;
- Software Instalado: software necessário na máquina que vai receber o agente.

Quando o MobiA recebe os relatórios de todas as máquinas, este está pronto para distribuir os agentes pelas mesmas, de forma equilibrada. Então, quando um agente requisita a máquina para qual se deve mover, juntamente com a mensagem também é enviada a lista dos seus requisitos. Com a lista de requisitos do agente confrontada com as características das máquinas através dos relatórios, o MobiA efetua a sua decisão. Para isso, existe um ciclo que percorre todos os agentes para verificar onde estes podem ser mantidos. Caso o agente tenha só um destino, este é logo movido. Caso contrário, é enviado para a máquina com menor número de agentes. Sempre que é recebido um novo pedido para mover um agente, é pedido um novo relatório (atualizado) com o estado das máquinas.

3.3.4 Simulação

O início do processo de simulação dá-se quando todos os ScenA notificam o agente gestor do sistema que estão prontos para iniciar a mesma. A Figura 3.12 demonstra o desencadeamento de comunicações entre agentes quando o TMA solicita o início de execução.

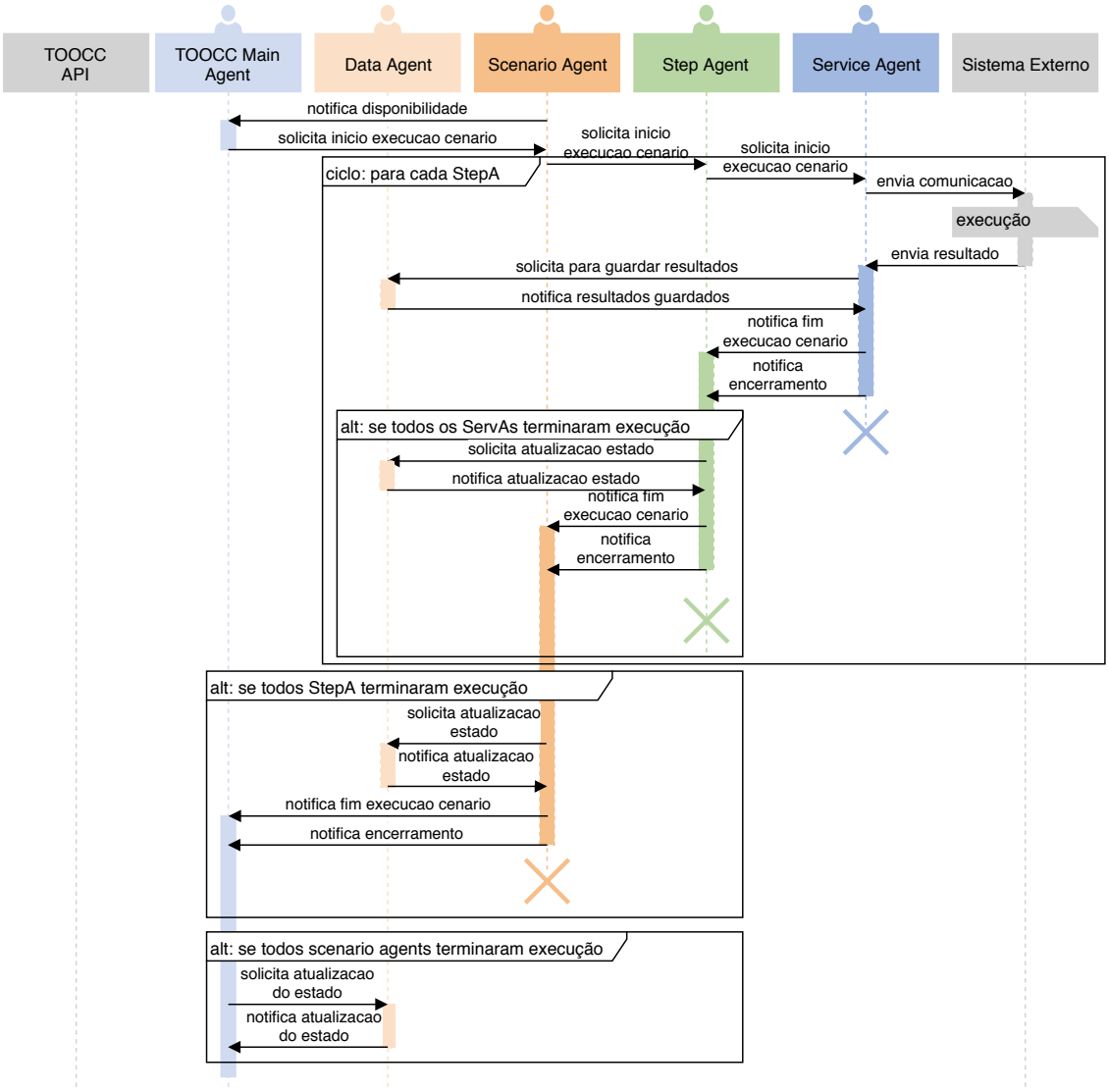

Figura 3.12: Diagrama de Sequência UML relativo à perspectiva geral das comunicações entre agentes para a simulação

Assim que um ScenA recebe o pedido para iniciar a simulação, este verifica qual é o StepA que deve iniciar a sua execução, uma vez que é necessário transferir o resultado de uma fase de execução, para a fase seguinte. De seguida, o StepA solicita a todos ServAs anteriormente criados para que possam então estabelecer a comunicação com os serviços externos. Quando é recebido o resultado da execução, é pedido ao DataA para atualizar o estado da simulação. Para além disso, também notifica o StepA que o criou que já terminou a sua tarefa e que vai encerrar-se. Por sua vez, quando o StepA recebe o resultado do serviço este armazena-o para que possa ser utilizado pela fase seguinte. Caso necessário, e se já todos os seus ServAs já estiverem encerrados, vai proceder à solicitação da alteração do seu estado de execução e comunicar ao ScenA que já terminou a realização da sua tarefa e que vai encerrar. O ScenA analisa qual é o próximo StepA a executar e repete o processo até todos eles concretizarem os seus objetivos. O processo é semelhante para o ScenA, onde cada vez que um StepA termina é verificado se foi o último, e caso seja, é notificado ao TMA o fim da execução e encerramento do ScenA. O TMA possui na sua base de conhecimento uma lista com

a simulação e cenários que foram criados. Assim que é informado pelo último ScenA, vai atualizar a informação da execução através do DataA. Assim, o resultado final é comunicado à API para poder ser mostrado ao utilizador.

A integração entre o SMA e os restantes módulos é realizada através da API TOOCC. A API disponibiliza um serviço particular para comunicar com o SMA - `RequestTOOCCMAS` - que cria um agente apenas para estabelecer a comunicação entre ambos, e que fica a aguardar o resultado. Durante o seu processo de escuta de novas mensagens, assim que o agente da API TOOCC recebe uma mensagem cujo emissor é o SMA principal, este vai verificar se tem a estrutura esperada de uma resposta do mesmo, e em caso afirmativo o serviço vai retornar a informação obtida.

3.4 Modelo de Dados

O modelo de dados da ferramenta TOOCC foi desenhado com três propósitos: o primeiro refere-se à necessidade de incutir nos modelos a utilização de dados reais e/ou simulados; o segundo destina-se à definição e preenchimento de modelos pré-definidos que servem de base à execução de serviços; e o terceiro diz respeito ao acompanhamento da simulação dos cenários.

Para o efeito, no âmbito deste projeto são consideradas duas fontes de dados, descritas nas secções seguintes: Repositório de Dados TOOCC (Secção 3.4.1) e Repositório de Dados GECAD (Secção 3.4.2).

3.4.1 Repositório TOOCC

O repositório TOOCC foi desenvolvido para dar suporte à configuração e execução dos cenários, e conta com uma API para a disponibilização e manipulação da informação nele contida. Além disso, durante a sua implementação foram considerados diferentes objetivos que resultam na implementação de bases de dados que procuram dar resposta a problemas distintos.

Processo de Simulação

O modelo de dados ilustrado na Figura 3.13 é referente ao armazenamento de informação relativa à simulação, e é utilizado como suporte à mesma atribuindo-lhe persistência.

Na Figura 3.13 é possível verificar que cada simulação - *Simulation* - pertence a um conjunto de simulações - *SetSimulation* - e é caracterizada por um conjunto de cenários que vão ser executados de forma paralela, nome, descrição, estado de execução (já totalmente executado ou não), tempo total da execução, e datas de criação e modificação. Para este caso é necessário ressalvar que este modelo já tem como perspectiva a evolução da ferramenta, onde é pretendido a configuração de simulações em simultâneo. Os cenários - *Scenario* - são compostos pelos mesmos campos de simulação, mas ao invés de possuir um conjunto de cenários, terá um conjunto de fases em que esse cenário irá executar, de modo sequencial, além de um campo a indicar quantas fases existem.

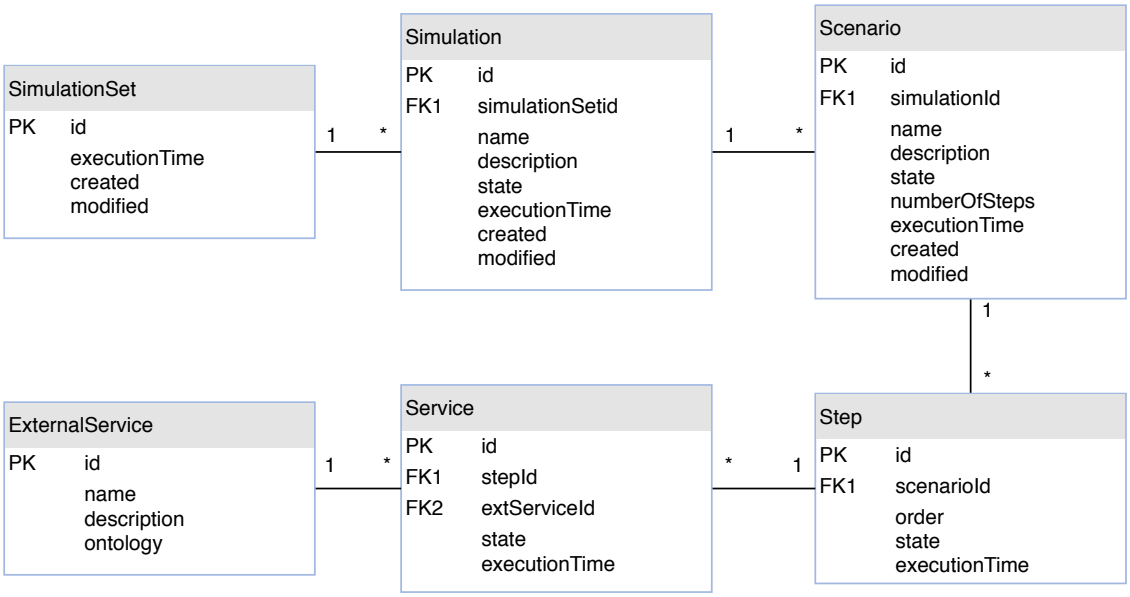

Figura 3.13: Modelo de dados UML para o processo de simulação

Cada passo de execução - *Step* - é constituído por uma lista de serviços que irá pedir a execução, um atributo que indica qual é a sua ordem de execução, estado de execução e tempo de execução. Estes serviços - *Service* - apenas são caracterizados pela indicação do serviço que vão executar, estado de execução e tempo de execução. Os serviços disponíveis para execução - *ExternalService* - é constituído por nome, descrição e a indicação da ontologia necessária para a comunicação, caso a utilize.

Programas de Demand Response

No entanto, a ferramenta TOOCC também pretende dispor de alguns modelos pré-definidos que podem ser utilizados pelos seus utilizadores para a execução de cenários mais específicos, como é o caso da DR. A Figura 3.14 demonstra o modelo desenhado para a execução de cenários para a criação e execução de programas DR.

Observando a Figura 3.14 é possível verificar uma grande diversidade de entidades e conceitos relacionados com programas de DR. Primeiramente é feita a configuração dos consumidores (tabela `Consumer`), que além dos campos de descrição como é exemplo o nome, possui vários perfis que o caracterizam na adoção de diferente tipos de programas DR, tais como: perfil geral de consumo; variação do consumo que o consumidor está disposto a praticar, ou seja, elasticidade; variação máxima do preço que está disposto a praticar por aplicação do programa; consumo iniciar disponível; contrato de electricidade adotado pelo consumidor; e uma lista de equipamentos - tabela `Device` - que o consumidor dispõe para aplicação dos programas. Os equipamentos variam entre diferentes tipos (ar condicionado, máquina de lavar, etc), contudo são normalmente caracterizados por aqueles cujo consumo pode ser manipulado, quer para redução, corte ou mudança de horário da sua utilização. Para além disso, cada equipamento ainda possui o seu próprio programa DR em que está inscrito, a informação do seu perfil de consumo e prioridade, onde quanto maior a prioridade, menos flexível a sua manipulação se torna.

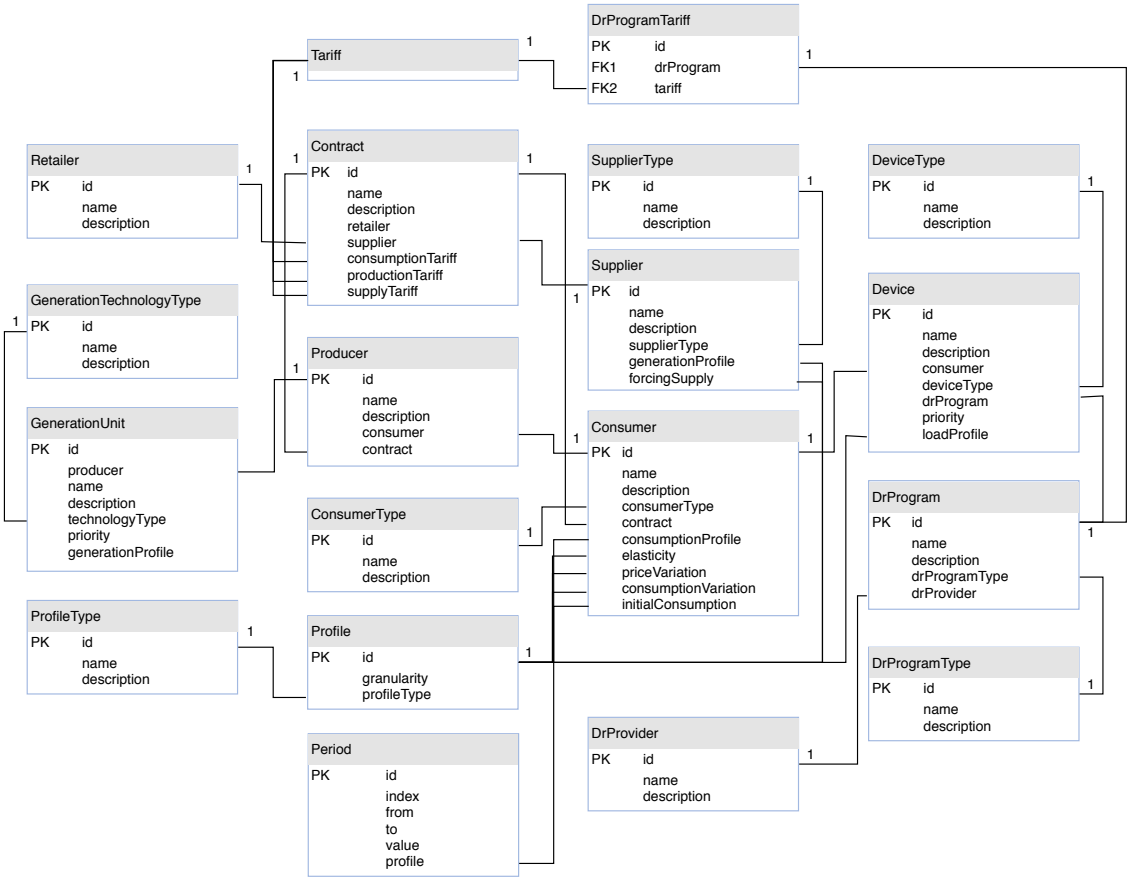

Figura 3.14: Modelo de dados UML para o modelo de criação de programas de *Demand Response*

Além dos consumidores também são considerados produtores - tabela `Producer` - especialmente de energia renováveis. No entanto, caso este seja na verdade um consumidor com capacidade de produção, denominando-se de *prosumer*, é necessário estabelecer essa ligação. Ademais, também possui um contrato com uma entidade de distribuição de energia - tabelas `Contract` e `Retailer`. Um produtor pode ter uma ou mais unidades de geração - GenerationUnit - que podem possuir diferentes tecnologias de geração (tabela `GenerationTechnologyType`), nomeadamente solar, eólica e outras. De forma equivalente aos dispositivos do consumidor, as unidades de produção também possuem um perfil de geração e prioridade.

Os perfis - tabela `Profile` - são definidos por um conjunto de períodos - tabela `Period` - que contêm um determinado valor dentro de um intervalo de tempo, definido por uma granularidade em minutos. Quando maior a granularidade, maior será o intervalo de tempo referente ao período.

Um contrato é determinado por um vendedor (`Retailer`), um fornecedor de energia para garantir o suprimento das necessidades dos contratantes - tabela `Supplier` - e as tarifas que são aplicadas ao fornecedor, consumidor e produtor. Para o fornecedor de energia é considerado o seu perfil de geração, ou seja, quanto é que ele pode fornecer por período de tempo. Neste modelo, também é possível forçar o fornecimento de energia em determinados períodos, de forma a efetuar estudos específicos.

Na caracterização de um programa DR é constituída pelo seu tipo - tabela `DrProgramType` - que pode ser de corte, redução ou *shifting*, o seu fornecedor - tabela `DrProvider` - e a sua tarifa de remuneração.

Tarifas

O modelo de tarifas possui dois objetivos: a criação de tarifas para o estudo do seu impacto nos cenários criados (eletricidade ou DR); e armazenamento de tarifas de reais. A Figura 3.15 apresenta o modelo de dados para a criação e armazenamento de tarifas.

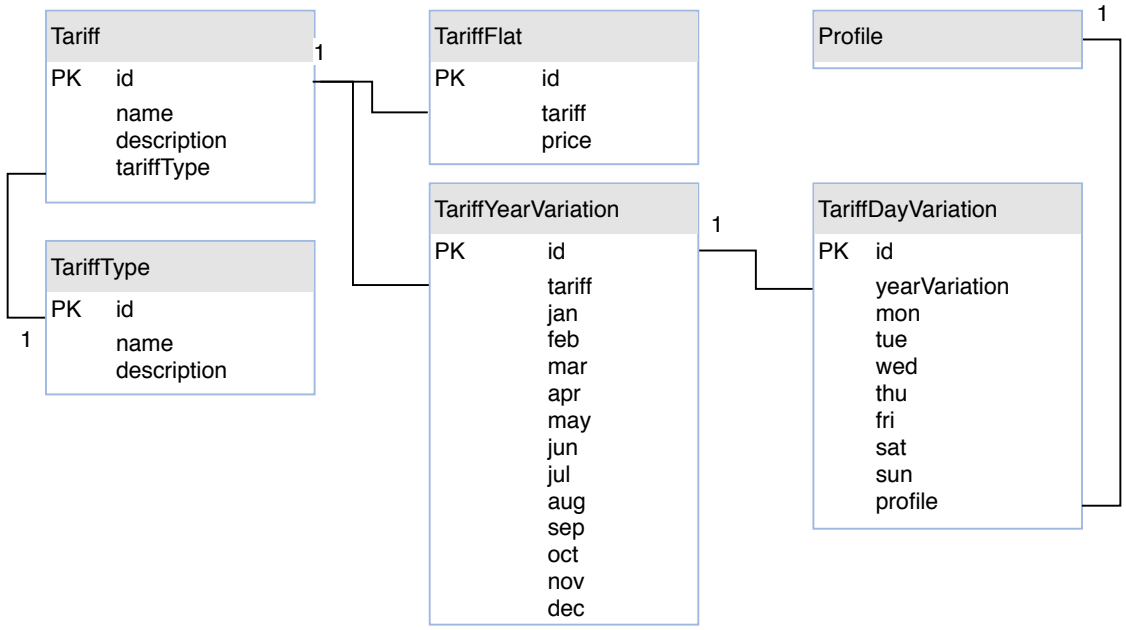

Figura 3.15: Modelo de dados UML para o modelo de Tarifação de Energia

Como foi dito anteriormente, o modelo apresentado acima pretende tanto armazenar tarifas reais como criar novas para efeito de simulação. Para efetuar a sua distinção, uma tarifa tem na sua caracterização um tipo de fonte (tabela `SourceType`), ou seja, real ou simulação. Além disso também possui um outro tipo - tabela `TariffType` - que indica se a tarifa em questão é destinada a consumidores ou produtores. Na sua constituição, as tarifas podem apresentar-se de duas formas, ou como sendo *flat*, ou seja, é praticado o mesmo preço a qualquer instante (tabela `TariffFlat`), ou variando ao longo do ano (tabela `TariffYearVariation`). As tarifas que variam durante o ano (normalmente época de inverno e verão), também podem variar consoante o dia da semana (tabela `tariffDayVariation`), onde no limite é praticado um preço diferente para cada dia da semana.

Atualmente, a ferramenta conta com mais de 200 tarifas reais, respeitantes a diversos países da UE, Estados Unidos da América e Brasil.

3.4.2 Repositório GECAD

De modo a atingir resultados mais fiáveis e realistas, a ferramenta TOOCC beneficia de um variado e extenso conjunto de dados disponibilizados pela base de dados do GECAD. Na verdade, tratam-se de diversos conjuntos de dados (*datasets*), distribuídos por diferentes bases de dados, relacionados com dados de consumo, produção, condições meteorológicas, dados captados através da leitura de sensores, ofertas de MEE, entre outros.

As instalações físicas do GECAD estão divididas em dois edifícios existentes no campus ISEP-IPP, nomeadamente os edifícios I e N. No caso particular do edifício N, foram instalados vários sensores, quadros de energia e PLCs com o propósito de extrair informação sobre o edifício e os seus ocupantes, além de paneis fotovoltaicos instalados no telhado. O edifício possui doze escritórios, dois WC, dois laboratórios e uma cozinha. Os dados são recolhidos através da leitura de sensores que fornecem informação sobre a ocupação de cada escritório, temperatura interna e externa, radiação solar, níveis de CO_2, humidade, luminosidade, entre outros, e são armazenados com uma granularidade de 5 segundos, em bases de dados SQL Server. Também é mantido o histórico do consumo energético de tomadas, aparelhos de climatização e luzes, com uma granularidade de 10 segundos de intervalo, assim como a produção obtida através dos painéis fotovoltaicos instalados.

Além destes, o ISEP-IPP disponibiliza uma API com dados extraídos de uma estação meteorológica, com informação sobre a temperatura externa, radiação solar, velocidade do vento e sua direção, entre outros. A granularidade dos dados é de 5 minutos de intervalo.

3.5 Modelo Semântico

Estabelecer interoperabilidade entre sistemas heterogéneos é uma tarefa complexa, pois além da troca de informação, também é necessário proporcionar a partilha do conhecimento e compreender o comportamento das suas funcionalidades. A inclusão da semântica nas comunicações vem permitir o completo entendimento do vocabulário partilhado por ambas as partes, bem como definir as relações entre estes conceitos, com a finalidade de estabelecer uma comunicação mais eficaz. Deste modo, não existe apenas a troca de informação entre os sistemas, mas também de conhecimento.

É com este propósito que o GECAD vem a desenvolver ontologias de domínio na área dos SEE para a comunicação entre os seus sistemas, onde cada uma delas define os conceitos relacionados com a ferramenta, e as suas relações. De facto, estas ontologias extendem outras ontologias, nomeadamente vocabulários e taxonomias, de maior dimensão no domínio dos SEE, que promovem a sua reutilização. Isto significa que existe uma ou mais ontologias de nível superior que determinam a taxionomia e axiomas do domínio de modo geral, permitindo o posterior desenvolvimento de ontologias mais concretas (com maior expressividade) que implementam as especificidades dos diferentes sistemas onde serão aplicadas. A Figura 3.16 apresenta parte das ontologias desenvolvidas pelo GECAD que visam a interoperabilidade entre sistemas. Estas pretendem abranger diversos domínios dos SEE, considerando MEE, apoio à decisão a participantes de mercado, gestão energética em redes, medições, sistemas multi-agente, algoritmos de otimização, entre outros.

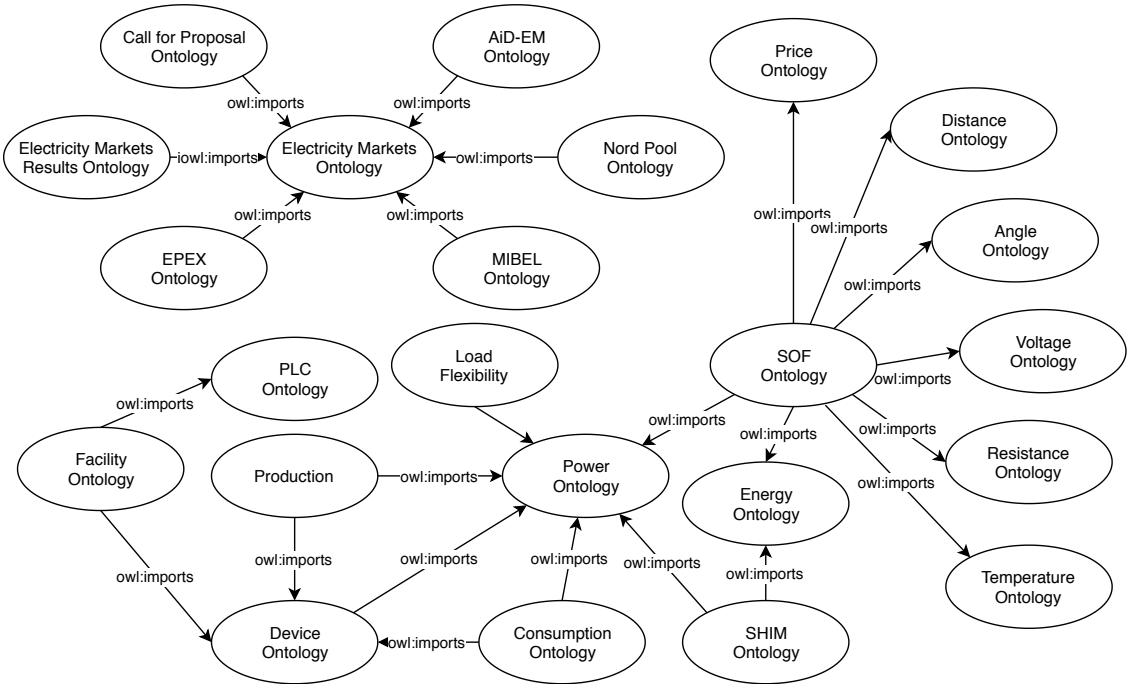

Figura 3.16: Ontologias de domínio para interoperabilidade

As ontologias foram desenvolvidas na linguagem Web Ontology Language - Description Logic (OWL-DL), com representação em RDF/XML e estão disponíveis publicamente através dos *websites* do MASCEM[1] e do repositório de ontologias para sistemas de energia inteligentes (Intelligent Energy Systems Ontologies)[2], para que estes serviços possam ser utilizados ou expandidos por outras aplicações. O Web Ontology Language (OWL)[3] é uma linguagem standard que nasceu de uma extensão do RDF, cujo objetivo prende-se com incutir mais expressividade e capacidade descritiva da lógica para a web semântica, tornando compatível a leitura tanto pelo homem como pela máquina. Por outro lado, o OWL-DL é uma sublinguagem desenhada para fornecer a máxima expressividade possível, mantendo a integridade computacional, decidibilidade e a disponibilidade de algoritmos de raciocínio práticos.

Por outro lado, embora estes serviços disponham de semântica com o propósito de assegurar a sua interligação, para que isso seja possível ainda é necessário que partilhem o mesmo conhecimento. A construção de uma ontologia apenas permite assegurar o entendimento dos conceitos que ela contém, ou que importa de uma outra ontologia. Para dois serviços serem capazes de se compreenderem é necessário que partilhem a mesma ontologia ou que seja criado um mapeamento de conceitos entre as suas respetivas ontologias. A ferramenta TOOCC faz uso da semântica para assegurar a co-simulação entre serviços, disponibilizados quer através de SMA ou serviços web. Grande parte destes serviços já são dotados da sua própria ontologia, contudo, também é permitida a inclusão de serviços que não estão dotados de semântica. Portanto, a TOOCC recorre ao uso das mesmas para estabelecer a comunicação, efetuando as traduções necessárias entre vocabulários, ou preencher a estrutura de dados necessária (no caso de não existir ontologia para comunicar com o serviço em questão).

[1] MASCEM Public Ontologies - `http://www.mascem.gecad.isep.ipp.pt/ontologies/`
[2] GECAD Intelligent Energy Systems Ontologies - `http://www.gecad.isep.ipp.pt/ontologies/ies/`
[3] OWL W3C - `https://www.w3.org/TR/owl-guide/`

3.5.1 Ontologia TOOCC

A Ontologia TOOCC (TCC), ao contrário das ontologias apresentadas até ao momento, é uma ontologia que procura descrever a nível aplicacional o modelo semântico da ferramenta TOOCC e é desenvolvida com o intuito de definir e descrever como é configurado um cenário, bem como facilitar o processo de interoperabilidade entre as ferramentas externas e posteriormente também a comparação de resultados. No desenvolvimento desta ontologia foi considerado um nível de abstração e flexibilidade que permita a sua evolução. Esta ontologia tem como sintaxe OWL DL, com representação em Turtle[4], através da utilização da aplicação *open-source* Protégé[5].

A Figura 3.17 ilustra as classes que definem a TCC, que se referem aos conceitos abordados neste trabalho que constituem a configuração, execução e análise de cenários.

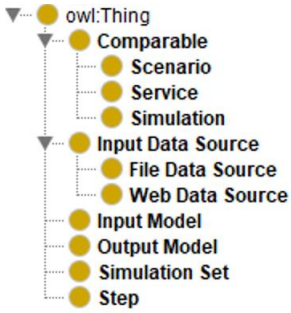

Figura 3.17: Classes da Ontologia TOOCC

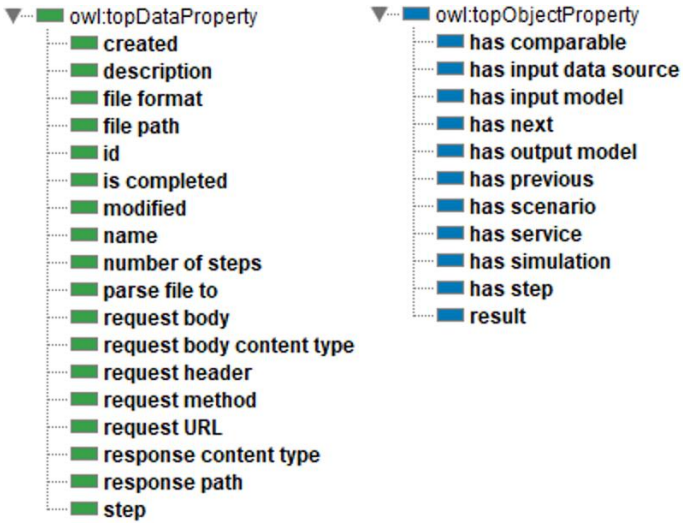

Figura 3.18: Propriedades dos Dados e dos Objetos da Ontologia TOOCC

As classes indicadas na Figura 3.17 identificam os conceitos envolvidos na execução da TOOCC, mas também consideram as restrições necessárias para assegurar o comportamento esperado do sistema da ferramenta, conjuntamente com as propriedades dos dados e objetos.

[4]Turtle - `https://www.w3.org/TR/turtle/`
[5]Protégé - `https://protege.stanford.edu/`

A Figura 3.18 ilustra as propriedades dos dados e objetos utilizados, identificados pelas cores verde e azul, respetivamente. Segundo a nomenclatura utilizada pela ferramenta Protégé, uma Propriedade do Objecto (ou *Object Property*) é a descrição da relação que existe entre dois objetos/classes. Por outro lado, uma Propriedade dos Dados (ou *Data Property*) procura representar a relação entre um objeto/classe e um valor literal, sendo que este último pode surgir sob vários tipos, tais como: texto; valor numérico; boolean; entre outros.

A Figura 3.19 apresenta a representação em UML da ontologia TCC,com a definição das suas classes, propriedades (Propriedades dos Dados) e relações (Propriedades dos Objetos).

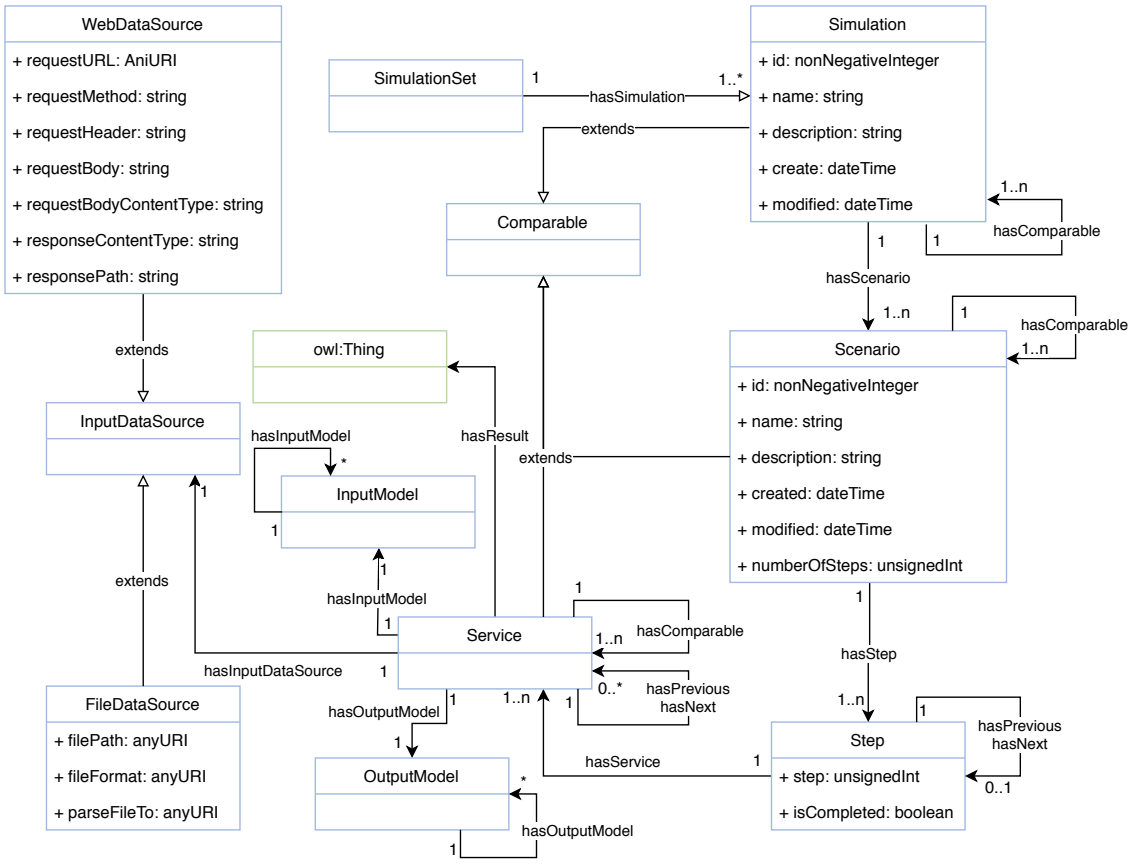

Figura 3.19: Ontologia TOOCC

Observando a Figura 3.19 é possível identificar as classes pertencentes à composição do modelo, juntamente com as suas propriedades e relações.

Para uma melhor compreensão dos conceitos relativos à ontologia TCC, as Tabelas 3.1, 3.2 e 3.3 apresentam a descrição da mesma recorrendo a sintaxe Description Logic (DL) (Baader et al. 2003), que permitem a representação do conhecimento. A expressividade da TCC em DL é $ALCQ(D)$, ou seja, permite: demonstrar linguagem atributiva (AL), que inclui negação atómica, interseção conceptual, restrições universais e quantificação existencial limitada; negação concetual complexa (C); restrições qualificadas de cardinalidade (Q); e a utilização de propriedades de tipos de dados e valores (D).

Tabela 3.1: Descrição das propriedades dos objetos da ontologia TOOCC recorrendo a sintaxe DL

Propriedades dos Objetos
$hasSimulation \sqsubseteq R$
$hasScenario \sqsubseteq R \quad \top \sqsubseteq \leq 1\, hasScenario$
$hasStep \sqsubseteq R \qquad \top \sqsubseteq \leq 1 hasStep$
$hasService \sqsubseteq R$
$hasPrevious \sqsubseteq R$
$hasNext \sqsubseteq R$
$hasInputModel \sqsubseteq R$
$hasInputDataSource \sqsubseteq R$
$hasOutputModel \sqsubseteq R$
$hasResult \sqsubseteq R$
$hasComparable \sqsubseteq R$

Tabela 3.2: Descrição das propriedades dos dados da ontologia TOOCC recorrendo a sintaxe DL

Propriedades dos Dados	
$id \sqsubseteq U$	$\top \sqsubseteq \leq 1 id$
$name \sqsubseteq U$	$\top \sqsubseteq \leq 1 name$
$description \sqsubseteq U$	$\top \sqsubseteq \leq 1 description$
$created \sqsubseteq U$	$\top \sqsubseteq \leq 1 created$
$modified \sqsubseteq U$	$\top \sqsubseteq \leq 1 modified$
$numberOfSteps \sqsubseteq U$	$\top \sqsubseteq \leq 1 numberOfSteps$
$step \sqsubseteq U$	$\top \sqsubseteq \leq 1 step$
$isCompleted \sqsubseteq U$	$\top \sqsubseteq \leq 1 isCompleted$
$filePath \sqsubseteq U$	$\top \sqsubseteq \leq 1 filePath$
$fileFormat \sqsubseteq U$	$\top \sqsubseteq \leq 1 fileFormat$
$parseFileTo \sqsubseteq U$	$\top \sqsubseteq \leq 1 parseFileTo$
$requestURL \sqsubseteq U$	$\top \sqsubseteq \leq 1 requestURL$
$requestMethod \sqsubseteq U$	$\top \sqsubseteq \leq 1 requestMethod$
$requestBody \sqsubseteq U$	$\top \sqsubseteq \leq 1 requestBody$
$requestBodyContentType \sqsubseteq U$	$\top \sqsubseteq \leq 1 requestBodyContentType$
$responseContentType \sqsubseteq U$	$\top \sqsubseteq \leq 1 responseContentType$
$responsePath \sqsubseteq U$	$\top \sqsubseteq \leq 1 responsePath$

Tabela 3.3: Descrição das propriedades das classes da ontologia TOOCC recorrendo a sintaxe DL

Propriedades das Classes
$Comparable \sqsubseteq \top$
$SimulationSet \sqsubseteq \top$
$InputDataSource \sqsubseteq \top$
$FileDataSource \sqsubseteq InputDataSource \sqcap 1\, filePath \sqcap 1\, fileFormat \sqcap 1\, parseFileTo$
$InputModel \sqsubseteq \top \sqcap \exists\, hasInputModel\, 1\, InputModel$
$OutputModel \sqsubseteq \top \sqcap \exists\, hasOutputModel\, 1\, OutputModel$
$Scenario \sqsubseteq Comparable \sqcap \exists\, hasComparable \geq 1\, Scenario \sqcap \exists\, hasStep \geq 1\, Step \sqcap 1\, id \sqcap 1\, name \sqcap 1\, description \sqcap 1\, created \sqcap 1\, modified \sqcap 1\, numberOfSteps$
$Service \sqsubseteq Comparable \sqcap \exists\, hasComparable \geq 1\, Service \sqcap \exists\, hasPrevious\, Service \sqcap \exists\, hasNext\, Service \sqcap \exists\, hasInputModel \leq 1\, InputModel \sqcap \exists\, hasOutputModel \leq 1\, OutputModel \sqcap \exists\, hasResult \leq 1\, \top \sqcap \exists\, hasInputDataSource \leq 1\, InputDataSource$
$Simulation \sqsubseteq Comparable \sqcap \exists\, hasComparable \geq 1\, Simulation \sqcap \exists\, hasScenario \leq 1\, Scenario \sqcap 1\, id \sqcap 1\, name \sqcap 1\, description \sqcap 1\, created \sqcap 1\, modified$
$Step \sqsubseteq \top \sqcap \exists\, hasService \geq 1\, Service \sqcap \exists\, hasPrevious \leq 1\, Step \sqcap \exists\, hasNext \leq 1\, Step \sqcap 1\, step \sqcap 1\, isCompleted$
$WebDataSource \sqsubseteq InputDataSource \sqcap 1\, requestURL \sqcap 1\, requestMethod \sqcap 1\, requestHeader \sqcap 1\, requestBody \sqcap 1\, requestBodyContentType \sqcap 1\, responseContentType \sqcap 1\, responsePath$
$Comparable \sqcap InputDataSource \sqcap InputModel \sqcap OutputModel \sqcap SimulationSet \sqcap Step = \bot$
$Service \sqcap Scenario \sqcap Simulation = \bot$
$FileDataSource \sqcap WebDataSource = \bot$

A classe *Comparable* permite a abstração das classes que são comparáveis entre si, tal como *Service*, *Scenario* e *Simulation*. A classe *Scenario* descreve o cenário definido pelo utilizador. Este é definido por: um *id* que é o identificador do cenário; *name* que define o nome do cenário em string; *description* que permite a descrição do cenário em string; *created* que se traduz na data e hora de criação do cenário em questão em dateTime; *modified* é a data e hora da última atualização do cenário em dateTime; *numberOfSteps* que identifica o número de fases configuradas no cenário em unsignedInt; e um conjunto de instâncias de *Step*, estabelecido através da relação *hasStep*.

O *Step* descreve a fase de execução de um cenário. A fase de execução permite ao sistema compreender quais são os serviços que irão executar em simultâneo. Um *Scenario* pode ter um ou mais *Step*s, sendo que cada *Step* possui uma relação de *hasPrevious* caso tenha uma fase anterior, e de *hasNext* para relacionar com a fase posterior. No caso de ser o primeiro *Step* configurado, ou seja, a primeira fase de execução, não existe relação *hasPrevious*. O mesmo acontece caso se trate do último *Step* configurado, onde não

existe uma relação *hasNext* definida com outro *Step*. Um Step tem configurado um ou mais *Service*, através da relação *hasService*. Além destes, ainda tem definida a propriedade *isCompleted* (iniciado com valor "falso") que indica quando um *Step* já terminou a sua execução, e a propriedade *step* que identificam o número descritivo da fase de execução.

O objeto *Service* define um serviço fornecido por um SMA ou um *webservice*. Analogamente ao *Step*, um *Service* também possui as propriedades *hasPrevious* e *hasNext*, que indicam o(s) serviço(s) que o antecede(m) e precede(m), respetivamente. Contudo, tanto os seus precedentes como os seguintes pertencem a fases de execução diferentes. Esta propriedade serve para criar uma precedência em que o resultado de um serviço vai servir de entrada ao seu serviço seguinte. Para além destas, são definidas as propriedades *hasInputModel* que indica qual o modelo de entrada de dados *InputModel*, *hasOutputModel* que define o modelo de dados de saída *OutputModel*, *hasResult* que define o resultado da execução do serviço, que pode resultar no modelo de qualquer outra classe, definido por *Thing* (a classe de topo de onde todos os conceitos extendem em OWL), e ainda uma definição da fonte de dados utilizada para os dados de entrada - *InputDataSource* - feita através da relação *hasInputDataSource*.

A classe *Simulation* descreve a simulação definida pelo utilizador. Uma simulação possui um ou mais cenários configurados (*Scenario*); um identificador único (*id*); nome (*name*); descrição (*description*); data e hora de criação (*created*); e data e hora de atualização (*modified*). A classe *SimulationSet* é definida como sendo a raiz no modelo do TOOCC, que considera as várias simulações consideradas pelo utilizador, resultando no conjunto das instâncias de (*Scenario*) criadas.

Como já foi referido, um *Service* tem um *InputModel* (modelo de entrada), que por sua vez possui um *InputDataSource*, ou seja, a fonte de dados responsável por fornecer os dados ao modelo de entrada, através da propriedade *hasInputDataSource*. Existem dois tipos de fonte de dados: ficheiros (*FileDataSource*) ou serviços *web* (*WebDataSource*), onde este último consiste maioritariamente no acesso a APIs. Para o caso de ficheiros é utilizada a classe *FileDataSource*. O *FileDataSource* é definido pelo endereço do ficheiro (*filePath*) em `xsd:anyURI` (um URI válido), pelo formato do ficheiro (*fileFormat*) em string; e o modelo semântico para o qual o ficheiro deve ser traduzido (*parseFileTo*) que corresponde a um `xsd:anyURI`. A classe *WebDataSource* também possui um `xsd:anyURI`, o endereço do pedido do serviço (*resquestURL*), o método utilizado (`POST`, `GET`, ou outro)(*requestMethod*), informação sobre o cabeçalho da mensagem (*requestHeader*), corpo do pedido (mensagem) (*requestBody*), o tipo de dados que o corpo da mensagem contém (texto, `JSON`, ou outro) (*requestBodyContentType*), o tipo de dados da resposta do serviço (*responseContentType*), e por fim o caminho da resposta, caso seja um `JSON` ou `XML` (*responsePath*). Um *InputModel* pode ser composto por um ou mais *InputModel*.

O objeto *OutputModel* caracteriza-se como sendo o modelo de saída de dados de um serviço (*Service*). Esta é uma classe abstrata que permite a definição recursiva de diversos *OutputModel*, com recurso à propriedade *hasOutputModel*. Ou seja, um *OutputModel* pode ser composto por outros.

3.5.2 Aplicação das Ontologias

Para proporcionar o uso da ontologia TCC e das restantes ontologias de domínio identificadas, foi utilizada a ferramenta *open-source* Apache Jena[6]. Esta ferramenta possibilita a concepção de código Java para a implementação de interpretadores e conversores das ontologias, para que possam ser usados pelo SMA nas fases de preparação da simulação, execução da simulação, e por fim a comparação de resultados.

Numa perspetiva geral, as ontologias de domínio apenas são usadas para comunicar com os sistemas que oferecem os serviços externos, enquanto que a ontologia aplicacional TCC define como é feito o processamento da informação durante a simulação. Para a troca de mensagens internas a estrutura utilizada é o JSON, de forma a tornar o sistema mais ágil. Contudo, o conteúdo trocado permanece com o cariz semântico.

Fase de Preparação da Simulação

A Figura 3.20 demonstra a iniciação do processo de simulação, em que é necessário criar os agentes e que estes iniciam a sua Base de Conhecimento Semântica (BCS), para que esta possa ser utilizada durante o processo de simulação.

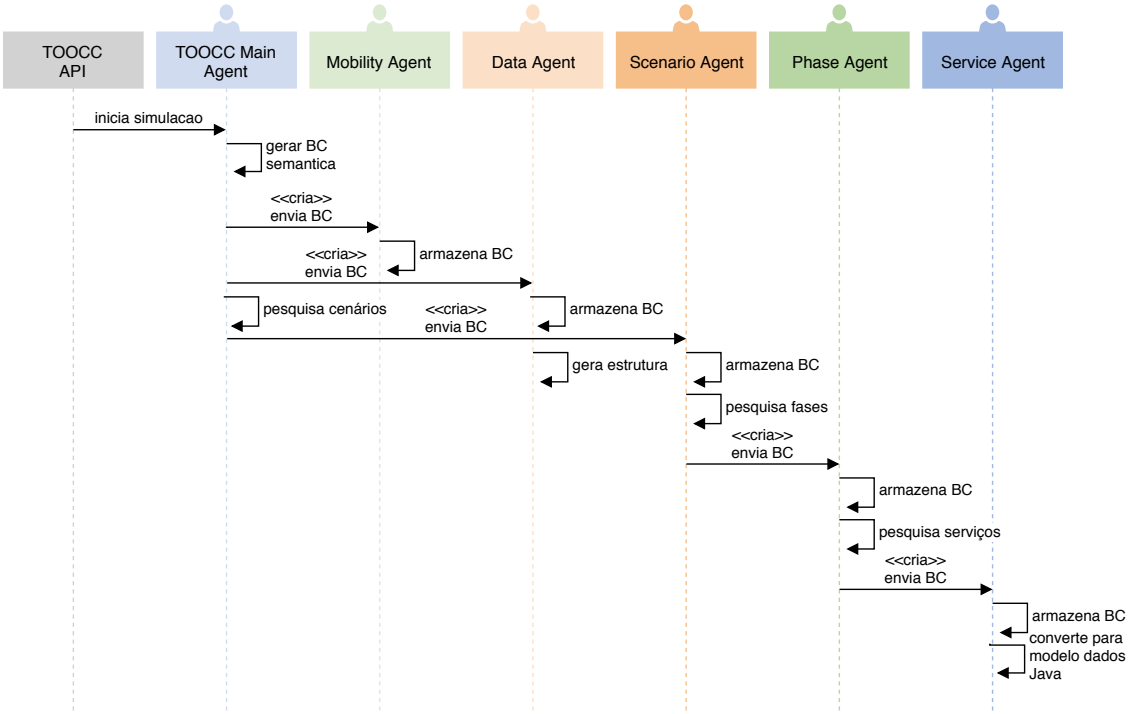

Figura 3.20: Comunicações com recurso a ontologias para a construção do SMA

Assim que o TMA recebe o pedido para iniciar a simulação por parte da TOOCC API, este tem de converter a informação fornecida para o modelo semântico utilizado pela ontologia TCC. A mensagem recebida é convertida para Turtle através de um conversor desenvolvido para o efeito. Todos os conversores e interpretadores usados para a manipulação de modelos

[6]Apache Jena - `https://jena.apache.org/`

dos agentes estão contidos num módulo desenvolvido para possibilitar a transformação de modelos em JSON, Java e semânticos, conforme a necessidade do sistema. A partir deste ponto é criada a BCS do agente. A BCS está incluída na BC do agente e é utilizada para obter informação sobre o comportamento que ele deve adotar.

Quando é obtido o modelo semântico correspondente à Figura 3.19, e caso seja a primeira vez que o agente vai executar, este cria os agentes MobiA e DataA, que recebem a mesma BCS. Posteriormente, o TMA procura obter uma lista com todos os cenários configurados através da utilização do Simple Protocol and RDF Query Language (SPARQL)[7]. Através dessa lista o agente motor é então capaz de criar os agentes ScenA e prosseguir com a execução da simulação. O SPARQL permite efetuar *queries* ao modelo semântico e assim extrair a informação pretendida. Apenas é enviado para cada ScenA o segmento da BCS que lhe diz respeito.

No caso do DataA, através do modelo semântico este é capaz de iniciar a criação da estrutura da simulação na base de dados, também recorrendo ao uso de SPARQL para inquirir quais as instâncias a criar. Esta base de dados é atualizada a pedido de cada agente envolvido na simulação. Por outro lado, o MobiA inicia-se e fica a aguardar os pedidos dos outros agentes para se moverem.

Assim que recebe a sua BCS, um ScenA vai utilizar o mesmo processo do TMA para obter uma lista de fases de execução que ele necessita de executar, e assim proceder à criação dos StepA. Também de forma análoga ao ScenA, um StepA vai inquirir à BCS que lhe foi transmitida quais são os serviços externos que vai requisitar durante a simulação, e assim proceder à criação dos ServA. No caso particular do ServA, logo após armazenar a sua BCS, tal como os outros agentes, para este ser capaz de estabelecer a comunicação com o serviço externo vai ter de efetuar a conversão do modelo semântico TCC para o seu próprio modelo Java. Este modelo é especialmente concebido para a comunicação com o serviço em questão, e tem o propósito de preencher todos os requisitos necessários para ocorrer a comunicação com o mesmo.

[7] SPARQL - `https://www.w3.org/TR/rdf-sparql-query/`

Fase de Simulação

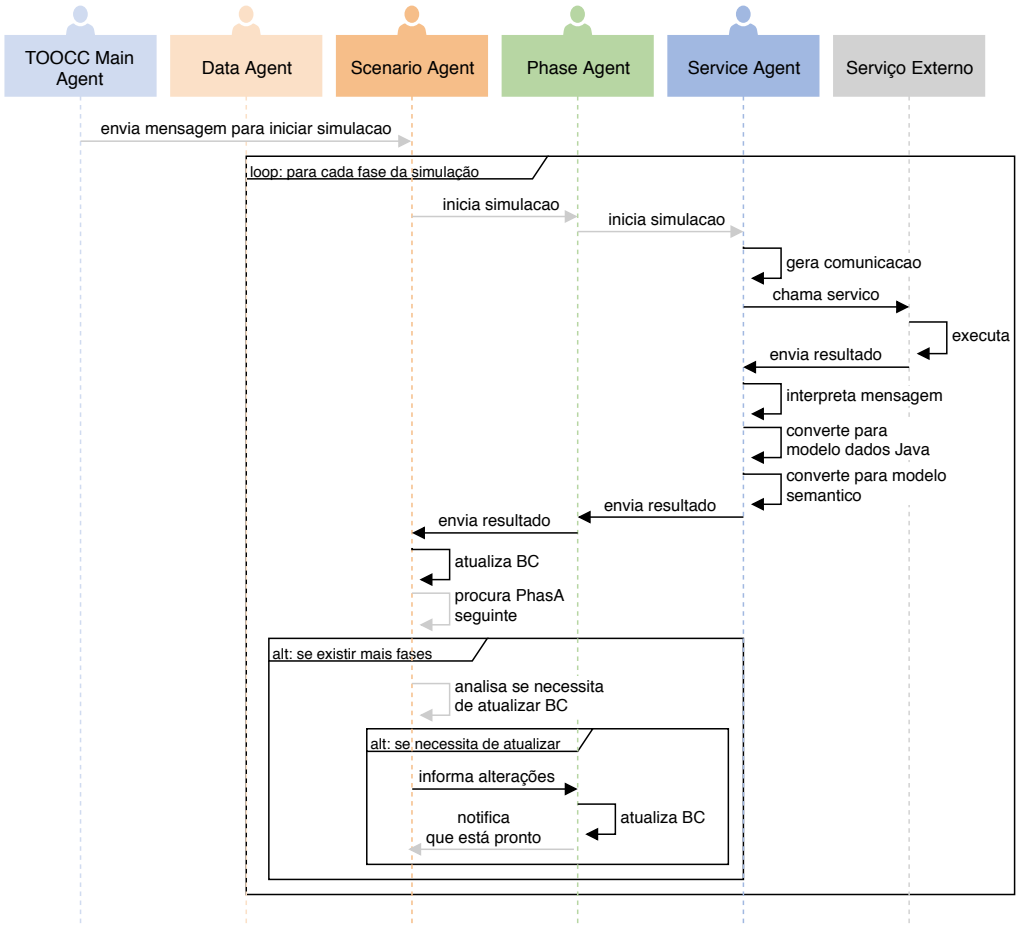

Figura 3.21: Comunicações com recurso a ontologias para processo de simulação

A Figura 3.21 ilustra como é realizada a simulação ao nível da utilização de semântica, nomeadamente sobre a atualização das BCS, conversão de modelos, e uso de ontologias de domínio externas.

O processo de simulação inicia-se assim que o TMA envia a todos os seus ScenA um pedido para iniciarem a simulação. Este pedido tem de ser analisado pelo ScenA, uma vez que este tem de apurar qual é o StepA responsável por iniciar a simulação, inquirindo através do SPARQL a sua BCS sobre qual StepA com (propriedade) step igual 1 e que não tenha uma precedência. Assim que é encontrado o StepA pretendido o pedido para iniciar a simulação é propagado até ao mesmo. Uma vez que os seus ServA executarão todos em simultâneo, o pedido será alargado a todos os seus ServA.

No momento em que é requerido o início de simulação ao ServA, e este já se encontra com o seu modelo de dados convertido em objetos Java, o agente procede para a construção da comunicação com o serviço externo. Para comunicar com o serviço o agente tem de analisar qual é o tipo de comunicação que é esperado, ou seja, se é através de um modelo semântico ou outro tipo de estrutura. Com a utilização de um conversor desenvolvido para

o caso, é então construída a mensagem para vai ser utilizada no corpo da mensagem (se for o caso).

De seguida é feita a chamada do serviço que é pretendido executar pelo ServA. Quando o resultado é retornado, o sistema vai verificar se a mensagem recebida corresponde ao formato esperado (a nível estrutural ou semântico). Caso esta condição se verifique, é necessário proceder novamente à conversão do resultado para o modelo de dados em Java, que por sua vez vai ser convertido para o modelo semântico, para que os dados possam ser enviados aos agentes que necessitam de atualizar a sua própria BCS, de modo a dar seguimento à simulação.

Uma vez que todos os ServA tenham executado, é necessário enviar os novos resultados ao ScenA pai, para que este possa dar continuidade à simulação. Sendo que cada StepA vai executar de forma sequencial, no momento em que um destes agentes termina, é necessário enviar os novos resultados ao StepA seguinte, para que este possa atualizar a sua BCS, bem como os seus ServA. Isto porque, segundo já foi descrito, alguns serviços externos vão receber como dados de entrada os resultados de serviços que executaram na fase anterior (através do StepA precedente) e consequentemente, aguardam a atualização dessa informação para iniciar o seu processamento.

Fase de Comparação de Resultados

A Figura 3.22 permite analisar como é efetuada a comparação de resultados. Esta tarefa é realizada assim que terminam a execução completa de todos os ScenA, que é quando já estão disponíveis todos os resultados.

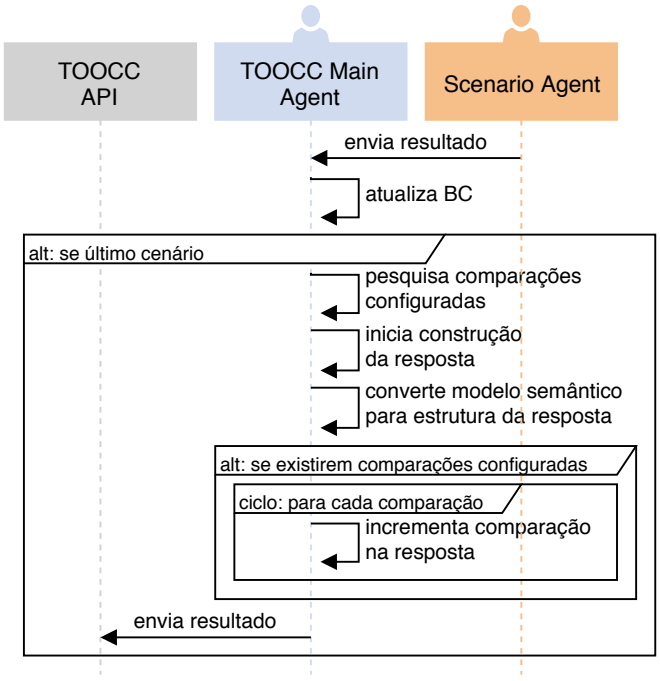

Figura 3.22: Comunicações com recurso a ontologias para o processo de comparação de resultados

A comparação de resultados recorre à classe *Comparable* do modelo semântico para estabelecer a correspondência entre resultados que são comparáveis. Apenas podem ser comparados serviços que tenham o mesmo modelo de saída de dados. Isto significa que quando dois serviços possuem entre si uma relação *hasComparable*, estes possuem o mesmo modelo de dados de saída.

Sempre que o TMA recebe um resultado por parte de um ScenA, este vai atualizar a sua BCS. Se todos os resultados de todos os ScenA, já tiverem sido retornados, o TMA vai iniciar a correspondência de resultados. Primeiramente vai obter as comparações configuradas. De seguida, inicia a construção da resposta, iniciando a conversão dos resultados do modelo semântico para o formato pretendido na comunicação com a TOOCC API. Caso tenham sido encontradas comparações, estas vão começar a ser analisadas. Para cada comparação encontrada será feita uma relação direta e adicionado o resultado à resposta principal, já com a estrutura pretendida. No final, a mensagem é enviada ao agente em execução na TOOCC API.

3.6 Interface Gráfica do Utilizador

Nesta secção é descrita a GUI, que consiste no módulo que estabelece a ponte entre o utilizador e o sistema.

Um dos principais objetivos deste componente prende-se com a necessidade de disponibilizar ao utilizador uma ferramenta para montar e configurar cenários de forma flexível, para que este não tenha de reprogramar lógica do cenário cada vez que deseja fazer alterações no mesmo. Isto significa que, através do uso desta ferramenta, o utilizador pode adicionar ferramentas, dados, e efetuar outras alterações, sem a necessidade de alterar código

Muitas das ferramentas e serviços disponíveis para a utilização da TOOCC carecem de uma grande quantidade de tempo e atenção por parte do utilizador. Para utilizar estas ferramentas é necessário preencher modelos de dados que, além de incluir diferentes conceitos, podem considerar problemas em grande escala. Ao permitir a interoperabilidade entre serviços, o trabalho de configuração é acrescido, pois também é necessário definir a forma como os serviços se interligam. A realização de tarefas com este tipo de complexidade pode resultar na introdução involuntária de erro humano, que influenciam diretamente os resultados obtidos.

Assim, considerando os requisitos do sistema, é disponibilizada ao utilizador uma interface gráfica que permite ao utilizador o acompanhamento do processo de simulaçao, que considera a construção de cenários, configuração dos mesmos, a simulação e a análise de resultados. Esta interface é disponibilizada através da rede interna do GECAD, garantindo a segurança da execução de serviços que atuem nas instalações físicas.

A escolha da linguagem de programação deste módulo foi feita segundo a tendência do tipo de ferramentas utilizadas no GECAD. Nos últimos anos, tem existido um maior investimento na utilização de interfaces gráficas que possam ser multiplataforma e facilmente acessíveis do exterior da rede da instituição onde se encontra (ISEP-IPP). Isto permite facilmente efetuar demonstrações de projetos ou de outros trabalhos a partir de qualquer lugar. Por este motivo, optou-se pela elaboração de uma interface gráfica *web*. Para a linguagem, a escolha recaiu sob a Linguagem de Marcação de Hipertexto (HTML)[8] com recurso ao *Javascript (JS)*[9],

[8] W3Schools HTML: https://www.w3schools.com/html/
[9] *Javascript* Página *Web*: https://www.javascript.com/

que é amplamente usado para o efeito. Além disto, para permitir o desenvolvimento de aplicações mais robustas, é utilizada a biblioteca *JQuery*[10] do JS.

Internamente, a GUI contém um conjunto de controladores que estabelecem a comunicação entre o modelo, a aplicação *web*, e os restantes módulos.

A Figura 3.23 apresenta a página inicial de acesso à ferramenta, onde é possível visualizar o seu logótipo.

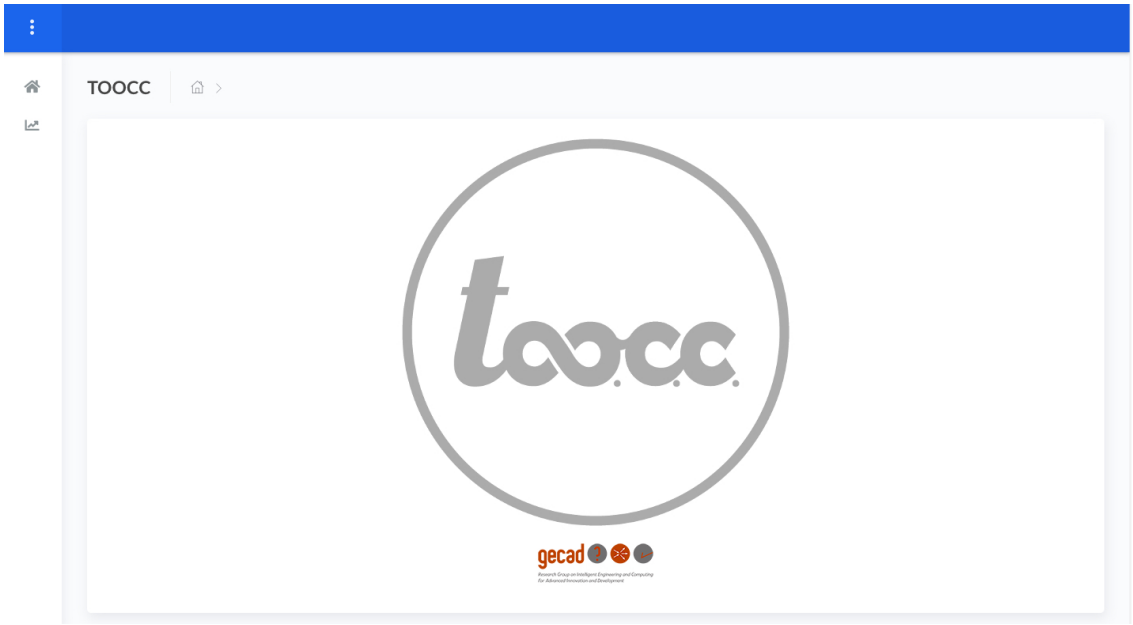

Figura 3.23: Página inicial

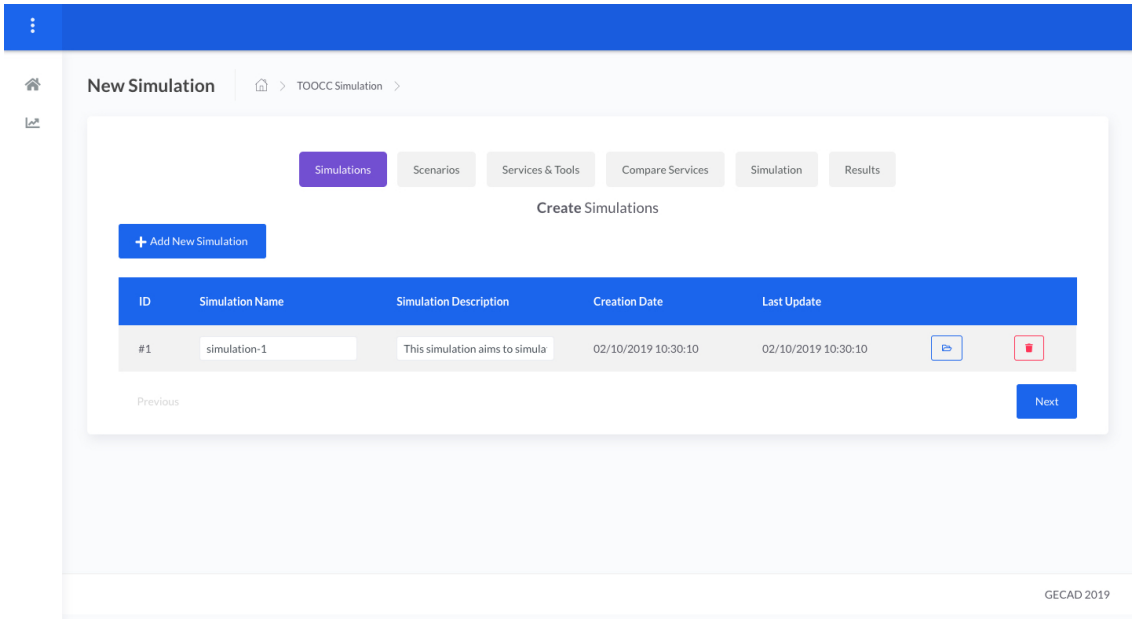

Figura 3.24: Página de criação de simulações

[10] *JQuery* Página *Web*: https://jquery.com/

Através do menu lateral é possível aceder ao menu responsável por dar início à configuração das simulações, como é representado na Figura 3.24. O processo de configuração é realizado através de passos, que são visíveis na parte superior da página. Cada passo possui um objetivo bem definido, e só é possível avançar no processo quando o passo atual é concluído. Contudo, é possível voltar no processo para efetuar alterações. A TOOCC permite a criação de diferentes simulações em simultâneo. Nesta página, é possível proceder à criação das simulações, indicando qual é o nome da mesma e a sua descrição. Assim que existe pelo menos uma simulação nomeada, é possível dar continuidade ao processo de configuração.

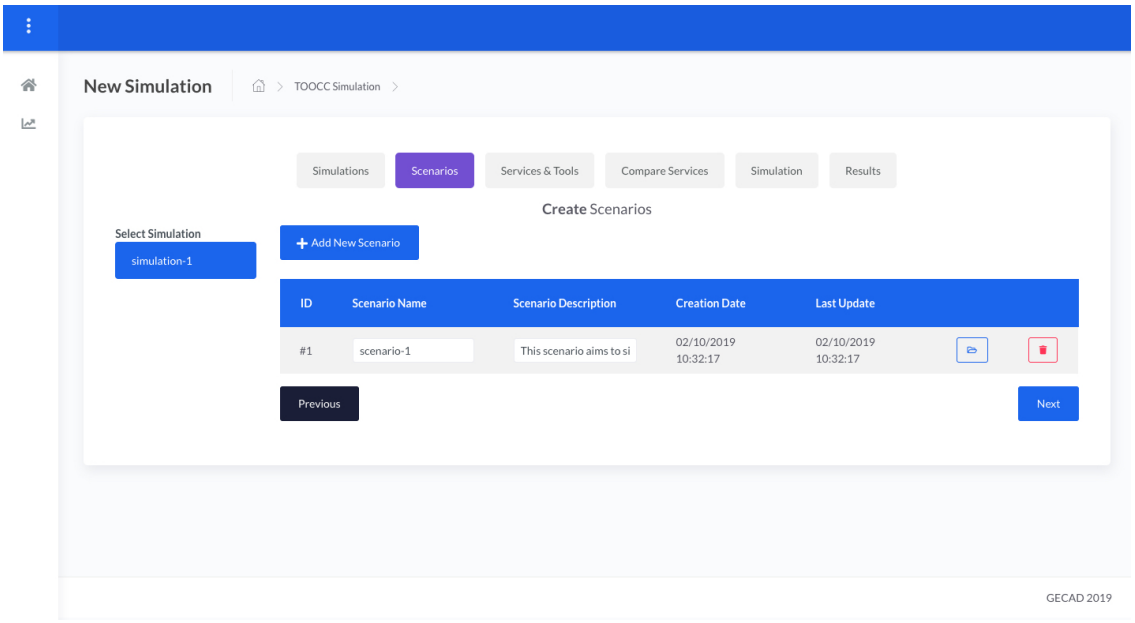

Figura 3.25: Página de criação de cenários

A Figura 3.25 ilustra o passo responsável por configurar os cenários dentro de cada simulação. Para tal, existe uma lista lateral das simulações existentes. O cenário criado vai ser adicionado à simulação selecionada. Os dados de um cenário são equivalentes aos das simulações, sendo que é necessário indicar o nome do mesmo, e, se desejar, uma descrição. Só é possível prosseguir para o próximo passo quando todas as simulações possuem pelo menos um cenário atribuído.

De seguida, ocorre a escolha das ferramentas e a sua configuração, apresentada na Figura 3.26. Este é o passo mais complexo de todo o processo, e também é aquele que requer mais atenção do utilizador. Para todos os cenários existentes (disponíveis na lista lateral), devem ser escolhidas as ferramentas ou serviços que serão executados. Contudo, para ser possível interliga-los, é preciso indicar em que fase de simulação eles vão executar, preencher o modelo de dados de entrada, e preencher informação adicional. O utilizador pode configurar várias fases para o mesmo cenário, e cada fase pode ter diversos serviços (*web* ou SMA). É necessário preencher os modelos de entrada de cada serviço, a não ser que este não exista. O utilizador também pode atribuir um nome da sua preferência ao serviço escolhido, com a finalidade de identificar melhor o seu objetivo na simulação.

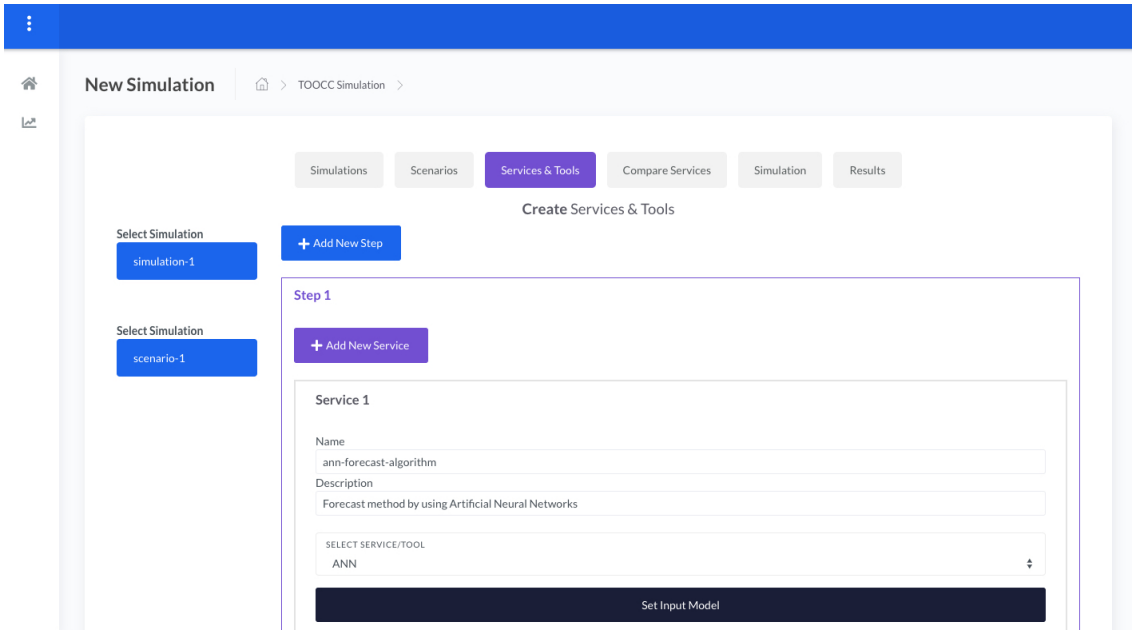

Figura 3.26: Página de seleção e configuração de serviços

A Figura 3.27 mostra o passo responsável pela configuração das comparações entre os dados. Essas comparações são apresentadas no momento de disponibilização dos resultados das simulações. Podem existir diversos serviços a serem comparados em simultâneo.

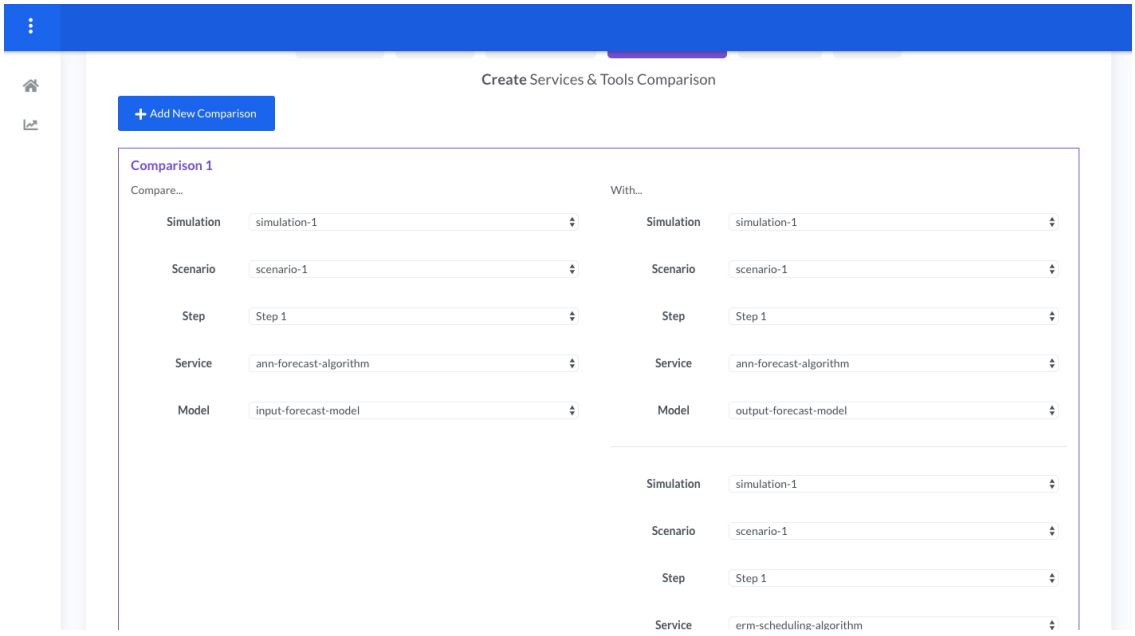

Figura 3.27: Página de configuração de comparações

A fase de execução é apresentada na Figura 3.28, onde é mostrada uma barra que indica o progresso da simulação.

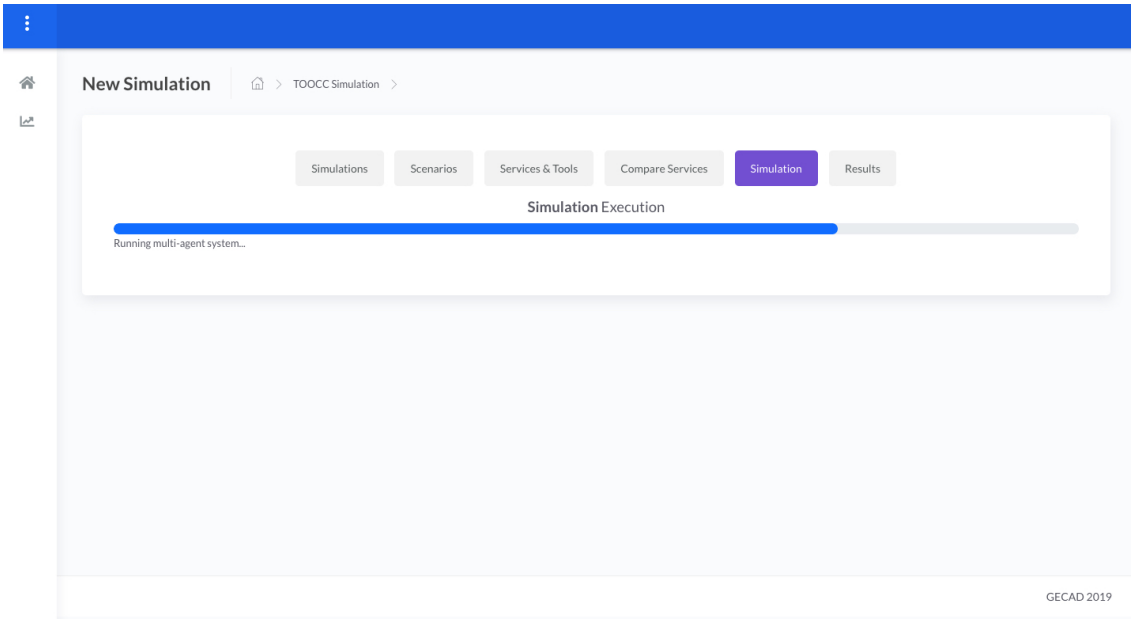

Figura 3.28: Página de execução de simulações

No final da execução de todos os cenários, são mostrados os resultados, como é ilustrado na Figura 3.29. Neste passo são apresentados todos os resultados das ferramentas, podem ser eles finais ou intermédios, e são também disponibilizados gráficos que demonstram as comparações configuradas anteriormente.

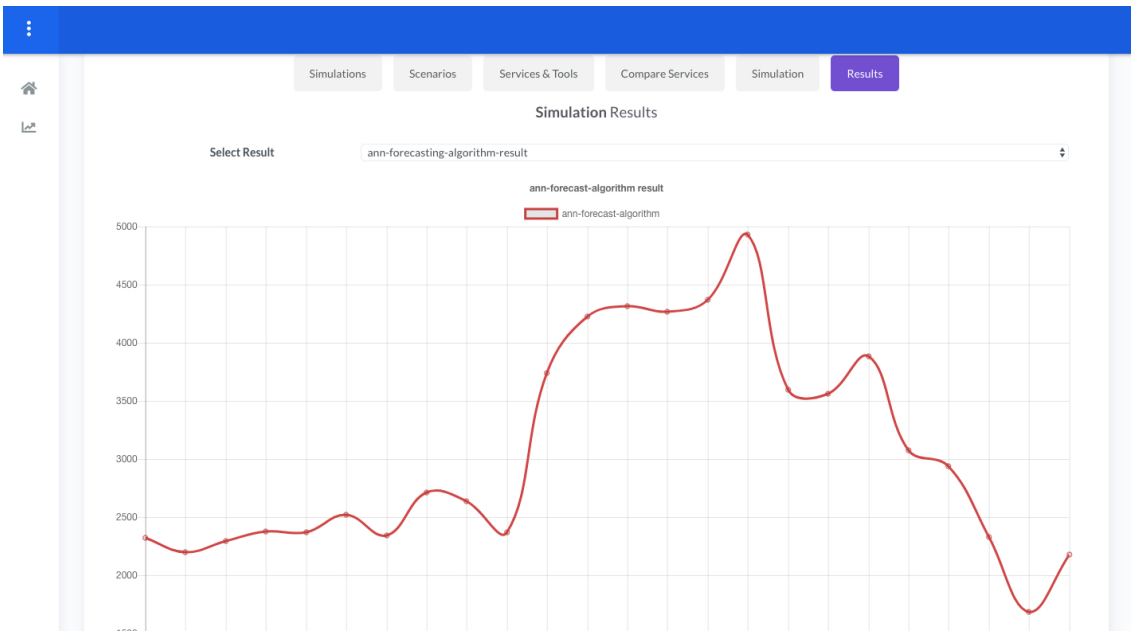

Figura 3.29: Página de visualização de resultados

Por último, é apresentada a construção de um modelo para execução de programas DR, que também faz parte deste simulador. Este modelo incorpora diversas entidades, com todas as suas informações e restrições.

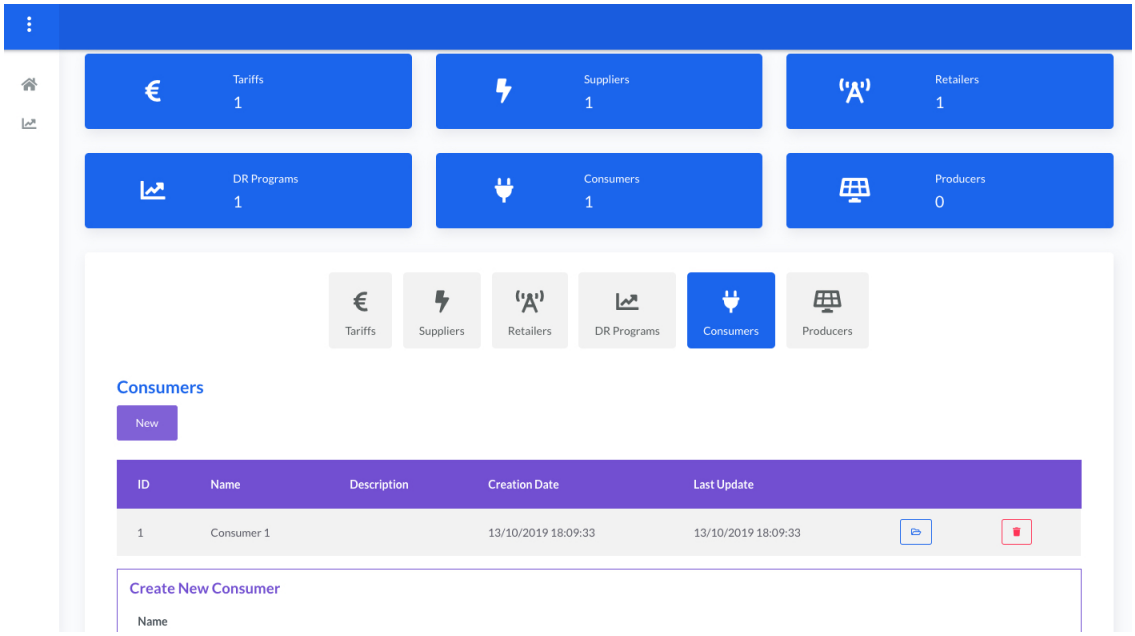

Figura 3.30: Página de visualização do modelo de DR

3.7 Considerações Finais

Este capítulo apresenta a *framework* TOOCC, como componente intermediário para estabelecer interoperabilidade entre ferramentas heterogéneas e configuração de cenários para estudo dos SEE. Com esta finalidade, a ferramenta age como facilitador na comunicação entre serviços externos, através da utilização de ontologias.

A interoperabilidade entre ferramentas e serviços tem como principal objetivo o reaproveitamento de diversos sistemas de simulação já existentes, agregando as suas capacidades individuais para o estudo de cenários que abordam problemas de domínio mais abrangentes. Contudo, a complexidade desta tarefa faz com que seja necessário adotar uma solução que além de garantir a comunicação entre os sistemas, também proporcione a escabilidade dos cenários, decomposição da complexidade no seu desenvolvimento, eficiência no processamento e modificações futuras.

Para garantir os objetivos da ferramenta, foram desenvolvidos módulos que ao interagirem entre si permitem com que o utilizador possa configurar cenários complexos de forma simples, e exponenciar a capacidade de processamento que este sistema requer. Assim, foi desenvolvido um SMA que através do uso de semântica é capaz de comunicar com serviços externos, quer disponibilizados por serviços *web*, quer por outros SMA. A semântica possibilita aos serviços a troca de conhecimento e a compreensão dos conceitos e relações presentes na comunicação.

Com o objetivo de ser capaz de interpretar mensagens através da utilização de semântica, a TOOCC recorre a dois tipos de ontologias: ontologias de domínio e aplicacionais. As ontologias de domínio pertencem aos serviços externos, e são disponibilizados por eles para que seja possível definir os dados e estrutura necessárias para a comunicação com os mesmos. Por sua vez, a ontologia aplicacional foi desenvolvida para determinar a forma como é executada a simulação e como os agentes devem agir.

Para a construção de cenários a ferramenta dispõe de uma aplicação *web* que, através de um processo faseado, permite ao utilizador escolher as ferramentas que deseja utilizar, recorrer a modelos pré-definidos, aceder a repositórios de dados de forma a enriquecer o cenário, configurar que resultados deseja comparar para o seu estudo, lançar a simulação, e analisar os resultados obtidos pelo sistema.

O Capítulo 4 apresenta os resultados obtidos para teste do funcionamento do sistema, através de casos de estudo concebidos para testar os diferentes módulos e funcionalidades.

Capítulo 4

Casos de Estudo

4.1 Introdução

O presente capítulo tem como objetivo a apresentação da avaliação da ferramenta TOOCC, proposta no âmbito desta dissertação.

Sendo a meta principal deste trabalho a configuração e execução de cenários que permitam a interoperabilidade entre serviços com características heterogéneas, para avaliação da ferramenta, foram considerados quatro casos de estudo que procuram analisar o comportamento do sistema e verificar se os seus objetivos são alcançados. Os casos de estudo foram desenhados para recriar situações em que as características e necessidades do sistema divergem, considerando diferentes fontes de serviços e grau complexidade na configuração e simulação dos cenários. Dependendo do nível de inteligência dos sistemas usados, estes vão executar de forma mais ou menos autónoma.

O primeiro caso de estudo pretende demonstrar a integração entre serviços *web*, onde é necessário efetuar o mapeamento entre os dados de saída de um serviço com os dados de entrada do serviço seguinte. De seguida é apresentado o segundo caso de estudo, que aborda a interoperabilidade entre SMA, nomeadamente dos simuladores MASCEM e AMBidS. O terceiro caso de estudo aborda a integração de SMA com serviços *web*, através da interoperabilidade entre o simulador MASGriP com serviços que disponibilizam as operações que os agentes terão de executar. Por último, o quarto cenário permite compreender como é feita a comparação de resultados e que vantagens esta funcionalidade pode trazer para o utilizador.

Além dos casos de estudo, durante o processo de desenvolvimento da ferramenta também foram efetuados testes funcionais a todos os módulos, bem como testes de integração. Adicionalmente, como já foi referido nas secções anteriores, houve uma especial atenção na adoção de metodologias que permitissem diminuir o tempo de execução, especialmente a latência causada pelas comunicações entre agentes e a interpretação das mensagens, que resultou na adoção de chunks.

4.2 Caso de Estudo 1 - Interoperabilidade Entre Serviços Web

No presente caso de estudo pretende-se demonstrar como a ferramenta TOOCC é capaz de estabelecer interoperabilidade entre serviços *web*, ou seja, Service-to-Service (S2S). Para tal, é considerado um cenário onde é aplicado um problema de DR, que tem como objetivo a redução dos custos operacionais e a remuneração justa dos recursos envolvidos.

4.2.1 Problema

Para este caso de estudo é considerada uma base de dados histórica com informação relativa ao mês de agosto de 2018 e com uma granularidade de 15 minutos de intervalo. Nela contém dados de consumo referentes a um agregador de uma rede elétrica que gere 144 consumidores com perfis variados (doméstico e industrial), 43 geradores de energia renovável (solar e vento), 1 fornecedor regular e 5 fornecedores adicionais, que são utilizados caso o primeiro (regular) não garanta a satisfação total de energia para alimentar a rede.

Este caso de estudo tem como objetivo conceptual a análise:

1. Da confiança e impacto económico dos modelos considerados;
2. Do impacto das interações e relações entre os atores dos modelos de negócio considerados;
3. Do impacto dos serviços desenvolvidos;
4. Do impacto de cada consumidor no conjunto de recursos agregados pelo sistema.

Para atingir os objetivos estabelecidos, primeiramente o agregador deverá efetuar o escalonamento energético da rede, levando em consideração todas as restrições existentes pelos seus utilizadores. Após o escalonamento, aqueles que possuem um contrato de DR, e que necessitam de reduzir o seu consumo para o que é estabelecido pelo agregador, devem ser recompensados pelo mesmo. Assim, é necessário apurar qual será o valor da remuneração justa para cada individuo. Para tal, é feita uma agregação baseada em função da quantidade de corte de energia. Por fim, para remunerar o utilizador final, vai ser apurada qual é a tarifa máxima de cada grupo, e essa será aplicada a todos os indivíduos nele contidos.

A redução de consumo é feita em certos dispositivos dos consumidores, que estão descriminados no seu contrato para aplicação da DR. Estes dispositivos podem ser aparelhos de climatização, tomadas, entre outros. Para garantir que um dispositivo sensível não é afetado é possível associar a cada dispositivo um grau de prioridade, em que o nível de prioridade máxima significa que deve ser cortado apenas em última necessidade, enquanto que aqueles com prioridade mais baixa podem ser considerados mais frequentemente.

4.2.2 Execução

Considerando uma perspetiva de configuração do cenário através da utilização da ferramenta TOOCC, para a sua execução vão ser considerados três serviços. O primeiro serviço será de escalonamento (otimização), que será seguido de uma agregação e, posteriormente, a execução de um algoritmo que determina a remuneração aplicável aos grupos. Os serviços devem executar de forma sequencial, onde parte dos dados de saída da primeira ferramenta são utilizados pela segunda, e assim sucessivamente. A Figura 4.1 ilustra uma perspetiva geral relativa ao caso e estudo, demonstrando quais os serviços que executam em cada fase e suas dependências.

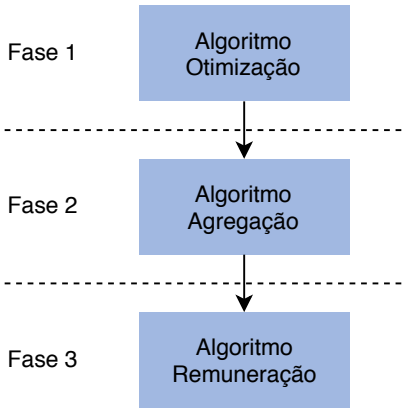

Figura 4.1: Perspetiva geral da execução do primeiro caso de estudo - S2S - pela ferramenta TOOCC

Quando o pedido de execução é recebido pelo SMA, a mensagem vai ser convertida para a BCS do agente. Por sua vez, este vai propagar a informação necessária para os agentes criados para a simulação. O excerto de código 4.1 apresenta a lista de ontologias importadas que a BCS deve considerar para a utilização dos serviços do cenário 1. Nele, é possível constatar que para este cenário são importadas as ontologias `TCC` e `MAT`, em que a primeira refere-se à ontologia utilizada para a representação da BCS, e a segunda é uma ontologia auxiliar para a presentação de dados em formato de matriz, que é muitas vezes esperado pelos serviços externos.

```
@base <http://www.gecad.isep.ipp.pt/> .
@prefix tcc: <ieso/tcc/> .
@prefix mat: <mat.ttl#> .
@prefix : <case-study-instantiation.ttl#> .

@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix owl: <http://www.w3.org/2002/07/owl#> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .

: rdf:type owl:Ontology ;
  owl:imports
    tcc: ,
    mat: ;
  .
```

Listing 4.1: Prefácio da Base de Conhecimento do cenário 1 (Turtle)

Para além da lista de ontologias importadas, a BCS também possui a descrição formal da composição do cenário, com a definição de quantas fases de execução existem. Tal como pode ser observado no excerto de código 4.2, o cenário 1 - `:scenario-1` - tem configuradas três fases (linhas 26 a 28). A informação de cada fase em particular vai ser transmitida ao StepA respetivo à execução de cada uma.

```
:simulation-set
  rdf:type
    owl:NamedIndividual ,
    tcc:SimulationSet ;
  tcc:hasSimulation
    :simulation-1 ;
```

```
7    .
8
9  :simulation-1
10   rdf:type
11     owl:NamedIndividual ,
12     tcc:Simulation ;
13   tcc:hasScenario :scenario-1 ;
14   tcc:id "simulation-1" ;
15   tcc:name "Service to Service (S2S) Simulation"^^xsd:string ;
16   tcc:descrition "The simulation demonstrates the Service to Service
       integration where the output of a Service is mapped to be the input
       of the next one."^^xsd:string ;
17   tcc:created "2019-07-25T16:26:24"^^xsd:dateTime ;
18   tcc:modified "2019-07-25T16:26:24"^^xsd:dateTime ;
19   .
20
21 :scenario-1
22   rdf:type
23     owl:NamedIndividual ,
24     tcc:Scenario ;
25   tcc:hasStep
26     :step-1-scen-1 ,
27     :step-2-scen-1 ,
28     :step-3-scen-1 ;
29   tcc:numberOfSteps "3"^^xsd:unsignedInt ;
30   tcc:id "scenario-1" ;
31   tcc:name "Service to Service (S2S) Scenario"^^xsd:string ;
32   tcc:descrition "The scenario demonstrates the Service to Service
       integration where the output of a Service is mapped to be the input
       of the next one."^^xsd:string ;
33   tcc:created "2019-07-25T16:26:24"^^xsd:dateTime ;
34   tcc:modified "2019-07-25T16:26:24"^^xsd:dateTime ;
35   .
```

Listing 4.2: Base de Conhecimento com a configuração geral do cenário 1 (Turtle)

A primeira fase a simular tem por referência `:step-1-scen-1` e está representada no excerto de código 4.3. Tal como mencionado anteriormente, esta diz respeito à execução do algoritmo de escalonamento `:service-optimization-algorithm`. A ligação entre o serviço que executa nesta fase, e o serviço de agregação que executará na fase seguinte é conseguido através da relação `tcc:hasNext`. O mesmo acontece com a classe `tcc:Step`, onde a ordem de execução é estabelecida também através da mesma relação, em conjunto com a propriedade `tcc:step`, que indica a ordem em que a fase vai executar. Adicionalmente, são construídas as relações para o seu modelo de dados de entrada `:InputOptimizationAlgorithm` e modelo de dados de saída `:OutputOptimizationAlgorithm`. Recorrendo a esta informação, o StepA também consegue determinar quais os ServA a criar. Posteriormente, a informação relativa ao serviço será transmitida ao agente ServA.

```
1  :step-1-scen-1
2    rdf:type
3      owl:NamedIndividual ,
4      tcc:Step ;
5    tcc:hasService :service-optimization-algorithm ;
6    tcc:hasNext :step-2-scen-1 ;
7    tcc:step "1"^^xsd:unsignedInt ;
8    .
9
```

```
:service-optimization-algorithm
  rdf:type
    owl:NamedIndividual ,
    tcc:Service ;
  tcc:hasNext :service-aggregation-algorithm ;
  tcc:hasInputModel :InputOptimizationAlgorithm ;
  tcc:hasInputDataSource :input-file-opti-algo ;
  tcc:hasOutputModel :OutputOptimizationAlgorithm ;
  .
```

Listing 4.3: Base de Conhecimento com perspetiva geral da configuração da primeira fase de execução do cenário 1 (Turtle)

O excerto de código 4.4 demonstra como é construído o modelo semântico de dados de entrada para o serviço de escalonamento/otimização. Neste modelo, é possível visualizar alguns dos campos necessários para a execução, nomeadamente o `:CutLimitIn`, `:ConsumptionIn`, `:ProductionIn`, entre vários outros. Estes campos transmitem informação ao algoritmo das características dos intervenientes da rede, para que este então possa efetuar um escalonamento da rede mais eficiente, baseando-se nos perfis de consumo e produção. Por outro lado, o excerto de código 4.5 mostra uma representação dos dados transmitidos em formato semântico.

```
:InputOptimizationAlgorithm
  rdf:type owl:Class ;
  rdfs:subClassOf
    tcc:InputModel ,
    [
      rdf:type owl:Restriction ;
      owl:onProperty tcc:hasInputModel ;
      owl:cardinality "1"^^xsd:nonNegativeInteger ;
      owl:onClass :CutLimitIn
    ] , [
      rdf:type owl:Restriction ;
      owl:onProperty tcc:hasInputModel ;
      owl:cardinality "1"^^xsd:nonNegativeInteger ;
      owl:onClass :ConsumptionIn
    ] , [
      rdf:type owl:Restriction ;
      owl:onProperty tcc:hasInputModel ;
      owl:cardinality "1"^^xsd:nonNegativeInteger ;
      owl:onClass :ProductionIn
    ] , [
      rdf:type owl:Restriction ;
      owl:onProperty tcc:hasInputModel ;
      owl:cardinality "1"^^xsd:nonNegativeInteger ;
      owl:onClass :DRConsumptionTariffIn
    ] ,
  ...
```

Listing 4.4: Versão adaptada da Base de Conhecimento com o modelo de entrada da primeira fase de execução do cenário 1 (Turtle)

```
# Optimization Algorithm Service Input Model for Cut Limit
:CutLimitIn
  rdf:type owl:Class ;
  rdfs:subClassOf mat:Matrix ;
  .

```

```
:iPmaxidr
        a         csi:CutLimitIn ;
        mat:item :iArray39iPmaxidr , :iArray33iPmaxidr , :iArray94iPmaxidr
      , :iArray2iPmaxidr , :iArray143iPmaxidr , :iArray42iPmaxidr , :
    iArray48iPmaxidr , :iArray100iPmaxidr , :iArray8iPmaxidr , :
    iArray76iPmaxidr , :iArray125iPmaxidr , :iArray134iPmaxidr , :
    iArray51iPmaxidr , :iArray24iPmaxidr , :iArray85iPmaxidr , :
    iArray15iPmaxidr , :iArray139iPmaxidr , :iArray14iPmaxidr , :
    iArray32iPmaxidr , :iArray67iPmaxidr , (...) .

:iArray39iPmaxidr
        a         mat:Array ;
        mat:item :iItem2316iArray39iPmaxidr , :iItem960iArray39iPmaxidr ,
    :iItem1558iArray39iPmaxidr , :iItem1487iArray39iPmaxidr , :
    iItem1828iArray39iPmaxidr , :iItem2046iArray39iPmaxidr , :
    iItem1381iArray39iPmaxidr , :iItem2174iArray39iPmaxidr , :
    iItem995iArray39iPmaxidr , :iItem1253iArray39iPmaxidr , :
    iItem1523iArray39iPmaxidr , :iItem429iArray39iPmaxidr , :
    iItem690iArray39iPmaxidr , :iItem1629iArray39iPmaxidr , :
    iItem1288iArray39iPmaxidr , :iItem124iArray39iPmaxidr , (...) .

:iItem2316iArray39iPmaxidr
        a         mat:Item ;
        mat:pos "2316"^^xsd:unsignedInt ;
        mat:val "0.0037"^^xsd:double .
```

Listing 4.5: Versão adaptada da Base de Conhecimento com um exemplo dos dados transmitidos no modelo de entrada da primeira fase de execução do cenário 1 (Turtle)

Observando o excerto de código 4.4, é possível visualizar que no seu modelo de dados de entrada está definida a classe `:CutLimitIn` como parte do mesmo. O código demonstra a definição dessa classe, onde é determinado que esta estende a classe pai `mat:Matrix`, definida pela ontologia `MAT`. Esta matriz possui um conjunto de itens que correspondem a vetores de valores. Todas as classes que são subclasses de `mat:Matrix` são definidas de igual forma, como são exemplos as classes `:ConsumptionIn` e `:ProductionIn`.

Após terminar a primeira fase de execução, o StepA já tem o conhecimento de qual é a fase que se segue. A segunda fase - `:step-2-scen-1` - consiste na execução do algoritmo de agregação, que vai criar grupos dos consumidores para determinar qual será a sua tarifa de remuneração mais adequada. No entanto, para que este serviço possa executar, ele necessita de preencher o seu modelo de dados de entrada `:InputAggregationAlgorithm` com alguns dos valores que compõem o modelo de saída de dados da fase anterior. A utilização dos valores entre o modelo de saída de um serviço para o modelo de entrada do serviço seguinte é efetuada através da definição da mesma classe. Um exemplo deste processo está ilustrado no excerto de código 4.6.

```
# Optimization Algorithm Service Output Model
:OutputOptimizationAlgorithm
  rdf:type owl:Class ;
  rdfs:subClassOf
    tcc:OutputModel ,
    [
      rdf:type owl:Restriction ;
      owl:onProperty tcc:hasOutputModel ;
      owl:cardinality "1"^^xsd:nonNegativeInteger ;
      owl:onClass :DGResOut
```

```
11      ] , [
12        rdf:type owl:Restriction ;
13        owl:onProperty tcc:hasOutputModel ;
14        owl:cardinality "1"^^xsd:nonNegativeInteger ;
15        owl:onClass :ReduceAmountResOut
16      ] , (...) ;
17    .
18
19  # Aggregation Algorithm Service Input Model for Optimization Solution In
20  :OptimizationSolutionIn
21    rdf:type owl:Class ;
22    rdfs:subClassOf
23      tcc:InputModel ,
24      [
25        rdf:type owl:Restriction ;
26        owl:onProperty tcc:hasInputModel ;
27        owl:cardinality "1"^^xsd:nonNegativeInteger ;
28        owl:onClass :ReduceAmountResOut
29      ] , [
30        rdf:type owl:Restriction ;
31        owl:onProperty tcc:hasInputModel ;
32        owl:cardinality "1"^^xsd:nonNegativeInteger ;
33        owl:onClass :DGResOut
34      ] ;
35    .
36
37  # Aggregation Algorithm Service Input Model
38  :InputAggregationAlgorithm
39    rdf:type owl:Class ;
40    rdfs:subClassOf
41      tcc:InputModel ,
42      [
43        rdf:type owl:Restriction ;
44        owl:onProperty tcc:hasInputModel ;
45        owl:cardinality "1"^^xsd:nonNegativeInteger ;
46        owl:onClass :OptimizationSolutionIn
47      ] , (...) ;
48    .
```

Listing 4.6: Versão adaptada da Base de Conhecimento com um exemplo da reutilização dos conceitos entre modelos de dados entre as fases de execução do cenário 1 (Turtle)

O código mostra que no modelo de dados de entrada referente ao algoritmo de agregação `:InputAggregationAlgorithm` (linha 38), está contida a classe `:OptimizationSolutionIn`. No que lhe concerne, esta consiste na agregação dos submodelos do `:OutputOptimization-Algorithm` que serão reutilizados pelo `:InputAggregationAlgorithm` (linha 46), ou seja, os dados relativos a `:ReduceAmountResOut` (linha 10) e `:DGResOut` (linha 15).

O excerto de código 4.7 apresenta uma visão geral da configuração do cenário para a execução do algoritmo de agregação `:service-aggregation-algorithm`. De forma equivalente à fase anterior, este serviço também tem na sua definição a indicação de qual será o serviço seguinte a ser executado, ou seja, o serviço de obtenção da remuneração dos consumidores `:service-remuneration-algorithm`.

```
:step-2-scen-1
  rdf:type
    owl:NamedIndividual ,
    tcc:Step ;
  tcc:hasService :service-aggregation-algorithm ;
  tcc:hasPrevious :step-1-scen-1 ;
  tcc:hasNext :step-3-scen-1 ;
  tcc:step "2"^^xsd:unsignedInt ;
  .

:service-aggregation-algorithm
  rdf:type
    owl:NamedIndividual ,
    tcc:Service ;
  tcc:hasPrevious :service-optimization-algorithm ;
  tcc:hasNext :service-remuneration-algorithm ;
  tcc:hasInputModel :InputAggregationAlgorithm ;
  tcc:hasOutputModel :OutputAggregationAlgorithm ;
  .
```

Listing 4.7: Base de Conhecimento com perspetiva geral da configuração da segunda fase de execução do cenário 1 (Turtle)

O modelo de entrada de dados do serviço de agregação está representado no excerto de código 4.8, enquanto que o excerto 4.9 tem uma representação do modelo de dados de saída `:OutputAggregationAlgorithm`.

```
:InputAggregationAlgorithm
  rdf:type owl:Class ;
  rdfs:subClassOf
    tcc:InputModel ,
    [
      rdf:type owl:Restriction ;
      owl:onProperty tcc:hasInputModel ;
      owl:cardinality "1"^^xsd:nonNegativeInteger ;
      owl:onClass :OptimizationSolutionIn
    ] , [
      rdf:type owl:Restriction ;
      owl:onProperty tcc:hasInputModel ;
      owl:cardinality "1"^^xsd:nonNegativeInteger ;
      owl:onClass :TariffIn
    ] ,
  (...)
  ;
  .
```

Listing 4.8: Versão adaptada da Base de Conhecimento com o modelo de entrada da segunda fase de execução do cenário 1 (Turtle)

```
:OutputAggregationAlgorithm
  rdf:type owl:Class ;
  rdfs:subClassOf
    tcc:OutputModel ,
    [
      rdf:type owl:Restriction ;
      owl:onProperty tcc:hasOutputModel ;
      owl:cardinality "1"^^xsd:nonNegativeInteger ;
      owl:onClass :BestKOut
    ] , [
```

```
        rdf:type owl:Restriction ;
        owl:onProperty tcc:hasOutputModel ;
        owl:someValuesFrom :AggregationListItem
      ] ;
    .
```

Listing 4.9: Exemplo do modelo de saída da segunda fase de execução do cenário 1 (Turtle)

Por último, é necessário proceder à execução da última fase, denominada por `:step-3-scen-1`. Aqui o objetivo é a execução do serviço `:service-remuneration-algorithm`, que faz a atribuição de uma tarifa de remuneração a cada entidade, de acordo com o grupo em que foi classificada. A configuração desta fase está demonstrada no excerto de código 4.10. Como é possível observar, ao contrário das fases anteriores, este serviço não possui a relação `tcc:hasNext`, já que é a última fase que será executada.

```
:step-3-scen-1
  rdf:type
    owl:NamedIndividual ,
    tcc:Step ;
  tcc:hasService :service-remuneration-algorithm ;
  tcc:hasPrevious :step-2-scen-1 ;
  tcc:step "3"^^xsd:unsignedInt ;
  .

:service-remuneration-algorithm
  rdf:type
    owl:NamedIndividual ,
    tcc:Service ;
  tcc:hasPrevious :service-aggregation-algorithm ;
  tcc:hasInputModel :InputRemunerationAlgorithm ;
  tcc:hasOutputModel :OutputRemunerationAlgorithm ;
  .
```

Listing 4.10: Base de Conhecimento com perspetiva geral da configuração da terceira fase de execução do cenário 1 (Turtle)

A BCS relativa ao modelo de dados de entrada da terceira fase está representada no excerto de código 4.11. Da sua observação é possível visualizar que o modelo tem na sua constituição o resultado da fase anterior através do triplo `:InputRemunerationAlgorithm tcc:hasInputModel :AggregationSolutionIn`, e o mesmo acontece para o `:Costin`. Além destes, ainda existem outros que se encontram encapsulados para efeito de demonstração.

```
:InputRemunerationAlgorithm
  rdf:type owl:Class ;
  rdfs:subClassOf
    tcc:InputModel ,
    [
      rdf:type owl:Restriction ;
      owl:onProperty tcc:hasInputModel ;
      owl:cardinality "1"^^xsd:nonNegativeInteger ;
      owl:onClass :AggregationSolutionIn
    ] , [
      rdf:type owl:Restriction ;
      owl:onProperty tcc:hasInputModel ;
      owl:cardinality "1"^^xsd:nonNegativeInteger ;
      owl:onClass :CostIn
```

```
    ] ,
  (..);
  .
```

Listing 4.11: Versão adaptada da Base de Conhecimento com o modelo de entrada da terceira fase de execução do cenário 1 (Turtle)

Finalmente, o excerto de código 4.12 apresenta o modelo de dados de saída do algoritmo para obter a remuneração. Este modelo é constituído por quatro submodelos (`:IndOut`, `:AvgOut`, `:MaxOut` e `:MaxIndOut`), que são o resultado final pretendido para o problema, e para o qual são convertidos os dados recebidos pelo ServA.

```
:OutputRemunerationAlgorithm
  rdf:type owl:Class ;
  rdfs:subClassOf
    tcc:OutputModel ,
    [
      rdf:type owl:Restriction ;
      owl:onProperty tcc:hasOutputModel ;
      owl:someValuesFrom :RemunerationListItem
    ] ;
  .

:RemunerationListItem
  rdf:type owl:Class ;
  rdfs:subClassOf
    mat:Item ,
    [
      rdf:type owl:Restriction ;
      owl:onProperty mat:item ;
      owl:cardinality "1"^^xsd:nonNegativeInteger ;
      owl:onClass :IndOut
    ] , [
      rdf:type owl:Restriction ;
      owl:onProperty mat:item ;
      owl:cardinality "1"^^xsd:nonNegativeInteger ;
      owl:onClass :AvgOut
    ] , [
      rdf:type owl:Restriction ;
      owl:onProperty mat:item ;
      owl:cardinality "1"^^xsd:nonNegativeInteger ;
      owl:onClass :MaxOut
    ] , [
      rdf:type owl:Restriction ;
      owl:onProperty mat:item ;
      owl:cardinality "1"^^xsd:nonNegativeInteger ;
      owl:onClass :MaxIndOut
    ] ;
  .
```

Listing 4.12: Exemplo do modelo de saída da terceira fase de execução do cenário 1 (Turtle)

4.2.3 Resultados

No final do processo de simulação do cenário os resultados são disponibilizados ao utilizador, para que este possa analisar e tirar conclusões sobre o mesmo. Estes resultados

compreendem tanto os resultados intermédios da simulação, que são fruto de cada fase de execução, como também os resultados finais, que são considerados como sendo aqueles que são extraídos da última fase de execução configurada.

As Figuras 4.2 e 4.3 apresentam os resultados obtidos do algoritmo de otimização.

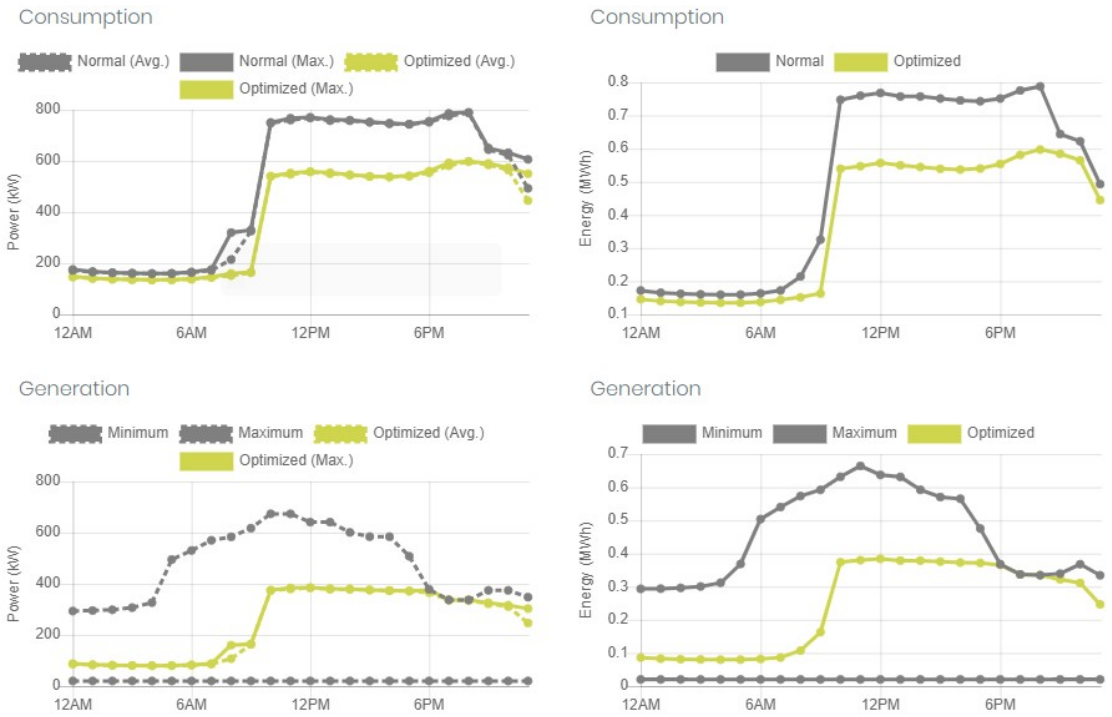

Figura 4.2: Resultado da otimização dos recursos dos consumidores e produtores para o cenário S2S

A Figura 4.2 demonstra que, no caso dos consumidores, o seu perfil de consumo após a otimização diminui mais significativamente entre as 10 AM e as 8 PM. Isto acontece porque este é o horário com mais flexibilidade para a redução por parte dos consumidores, sendo que muitas vezes estes encontram-se ausentes das suas residências. De uma outra perspetiva, a análise dos resultados dos produtores revela que, como o custo de geração era mais elevado, a percentagem considerada dessas tecnologias é inferior ao máximo que estas poderiam disponibilizar.

A Figura 4.3 ilustra os resultados obtidos para os fornecedores externos e indicação das necessidades energéticas que não foram satisfeitas.

Figura 4.3: Resultado da otimização dos recursos dos consumidores e produtores para o cenário S2S

Como é possível observar, a Figura 4.3 demonstra que, uma vez que o fornecedor adicional não atingiu o seu máximo no fornecimento de energia, não foi necessário recorrer aos fornecedores adicionais. Além disso, o facto de o fornecedor regular também não fornecer a sua capacidade máxima também justifica que o valor da energia não fornecida (ou non-supplier power) seja nulo, sendo que todos os consumidores foram abastecidos devidamente. Este processo evita o desperdício de recursos.

Na fase seguinte é executada a agregação dos consumidores em grupos (ou *clusters*), para determinar a remuneração adequada pela redução efetuada na fase anterior. Os centróides obtidos a partir do algoritmo possibilitam ao agregador efetuar uma estimação da média da potência reduzida, e classificar um recurso mediante os grupos criados. Este algoritmo permite testar a aplicação com diferentes *k clusters*. A Figura 4.4 indica a aplicação do método de agregação para a criação de 3 grupos, para a 32ª semana do ano.

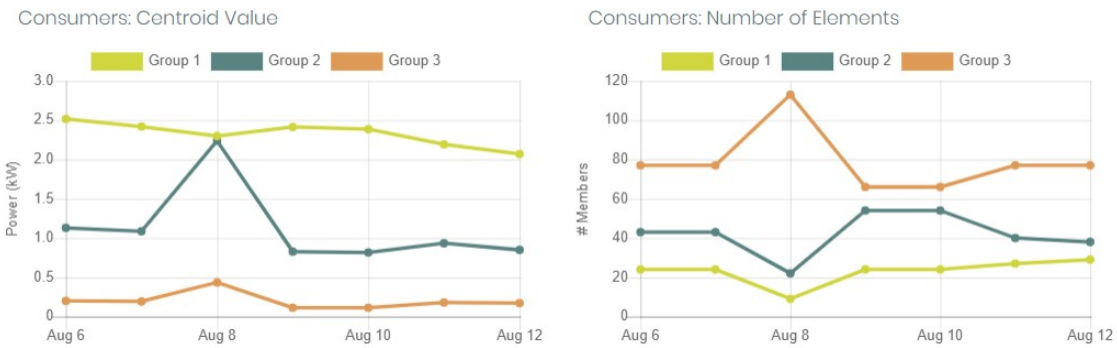

Figura 4.4: Resultado da agregação quando são criados 3 *custers* para o cenário S2S

Quando são considerados 3 grupos, verifica-se que a classificação resulta na colocação de 9 recursos no primeiro grupo, 22 no segundo grupo, e 113 no último grupo. Para o primeiro grupo, o seu centróide é aquele que apresenta uma maior redução, com cerca de 2.30kW, enquanto que o terceiro grupo apresenta uma menor redução, com valores que rondam os 0.43kW.

Por fim, são apresentados os resultados na fase de remuneração para um consumidor na Figura 4.5. Este algoritmo vai aplicar diferentes métodos de remuneração para que seja aplicado aquele que apresenta melhor custo/benefício para o agregador. O consumidor escolhido deriva dos resultados do estudo da criação de apenas 2 grupos, e é retirado especificamente do segundo grupo.

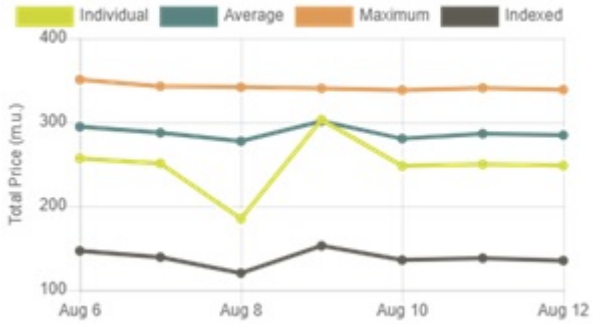

Figura 4.5: Resultado do algoritmo de remuneração de um consumidor para o cenário S2S

Existem quatro métodos de remuneração: individual, indexada, média e máximo. A tarifa indexada vai variar consoante o resultado de mercado, onde existe uma percentagem que pertence ao preço de mercado e a restante recorre à tarifa do consumidor. O método de remuneração máximo consiste em considerar a remuneração máxima no grupo. O método de remuneração média vai efetuar uma média das tarifas dos diferentes consumidores do grupo. E a tarifa individual é a própria tarifa do consumidor (para verificar se é vantajoso para o consumidor e se representa um maior incentivo à sua participação). No caso representado na Figura 4.5, a tarifa indexada tem uma atribuição de 30% para a tarifa fixa, e os restantes 70% para a tarifa indexada. É possível observar que a curva do método de tarifa indexada acompanha a variação da tarifa individual, incluindo os picos mínimos e máximos.

Para a atribuição da remuneração final, na perspetiva do agregador é importante que seja o menor valor possível, pois é ele que terá de pagar ao consumidor. Para este caso seria aplicada a tarifa indexada. Contudo, por este valor ser muito baixo pode não compensar ao consumidor a sua participação no programa e também fazer com que ele não tenha incentivo de voltar a participar no programa DR. O que se conclui é que a tarifa indexada deverá ter uma percentagem maior para a componente da tarifa fixa, que tem como base a tarifa aplicada ao consumidor. Desta forma o preço continua a ser apelativo para o agregador, mas também apresenta um valor mais justo para o consumidor.

4.3 Caso de Estudo 2 - Interoperabilidade Entre Sistemas Multi-Agente

O caso de estudo que se segue exemplifica a execução de um cenário que durante o seu processo de simulação estabelece interoperabilidade entre dois SMA, ou seja, MAS-to-MAS (MAS2MAS). Deste modo, o cenário consiste na integração das ferramentas de simulação MASCEM e ALBidS.

4.3.1 Problema

O objetivo deste cenário consiste em simular o modelo para o dia seguinte do mercado MIBEL, que representa o MEE presente em Portugal e Espanha, permitindo aos seus participantes (agentes) recorrer ao uso do simulador ALBidS para estabelecer o melhor negócio a fechar. Este cenário utiliza a informação de 110 participantes, em que desde o participante 1 ao 55 estão representados os compradores, e do 56 ao 110 são os vendedores de energia. Os dados utilizados são referentes ao dia 18 de fevereiro de 2015, e apresentam uma granularidade de 60 minutos de intervalo, ou seja, 24 períodos.

Adicionalmente, pretende-se perceber qual é o impacto que a utilização do sistema ALBidS traz, quer para o participante que solicita a sua ajuda, quer para o preço de mercado no geral. O simulador ALBidS permite ao seus utilizadores obter maior lucro.

4.3.2 Execução

Na configuração do cenário do caso de estudo MAS2MAS, é essencial ter em conta que estes simuladores já possuem inteligência própria. Isto significa que a única tarefa da ferramenta TOOCC consiste em iniciar os sistemas. Considerando este facto, são considerados três serviços: inicialização do simulador ALBidS, inicialização do simulador MASCEM, e serviço para iniciar a simulação do MASCEM. Não é necessário ordenar a execução do simulador ALBidS devido ao facto do MASCEM ter a capacidade de conversar diretamente com este sistema, logo que tenha conhecimento do seu endereço, que é fornecido através dos dados de entrada. A Figura 4.6 demonstra uma visão de como será configurado o cenário descrito.

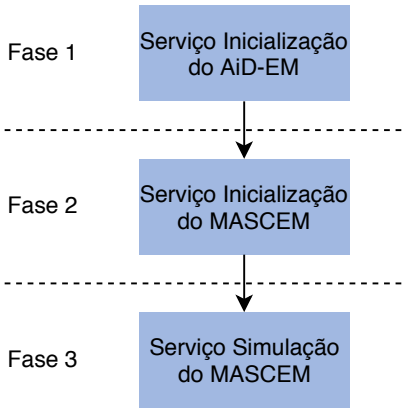

Figura 4.6: Perspetiva geral da execução do segundo caso de estudo - MAS2MAS - pela ferramenta TOOCC

Tal como acontece com a execução de todos os cenários, quando o SMA recebe a mensagem para iniciar a simulação, o TMA converte os dados transmitidos pelo ApiA para a BCS. O excerto de código 4.13 mostra as ontologias importadas para a construção da BCS, nomeadamente: a `TCC` que consiste na ontologia aplicacional desenvolvida; `EMO` que importa os conceitos relacionados com os MEE (Santos, Pinto, Vale et al. 2016); `EMR` que define os conceitos dos resultados do MEE (Santos, Pinto, Praça et al. 2016a); e `ADM`, que é a ontologia utilizada pelo ALBidS (Santos 2015).

```
1  @base <http://www.gecad.isep.ipp.pt/> .
2  @prefix tcc: <ieso/tcc/> .
3  @prefix emo: <http://www.mascem.gecad.isep.ipp.pt/ontologies/electricity
       -markets.owl#> .
4  @prefix emr: <http://www.mascem.gecad.isep.ipp.pt/ontologies/electricity
       -markets-results.owl#> .
5  @prefix adm: <http://www.mascem.gecad.isep.ipp.pt/ontologies/aid-em.owl
       #> .
6  @prefix : <case-study-instantiation.ttl#> .
7
8  @prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
9  @prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
10 @prefix owl: <http://www.w3.org/2002/07/owl#> .
11 @prefix xsd: <http://www.w3.org/2001/XMLSchema#> .
12
13 : rdf:type owl:Ontology ;
14   owl:imports
15     tcc: ,
16     emo: ,
17     emr: ,
18     adm: ;
19   .
```

Listing 4.13: Prefácio da Base de Conhecimento do cenário 2 (Turtle)

O excerto de código 4.14 apresenta o início da configuração do cenário, que permite ao TMA iniciar os ScenA necessários. Como é possível observar, o SMA vai executar apenas uma simulação (`:simulation-2`) que é composta por um único cenário. O cenário tem por nome `:scenario-2`, e é subdividido em três fases de execução (visível nas linhas 36 a 38). Através da análise à BCS, o ScenA possui a informação necessária para a criação dos

seus StepA. É criado um StepA por cada fase de execução, e a cada um deles é difundido o segmento da BCS que permite a sua configuração.

```
:simulation-set
  rdf:type
    owl:NamedIndividual ,
    tcc:SimulationSet ;
  tcc:hasSimulation
    :simulation-2;
  .

:simulation-2
  rdf:type
    owl:NamedIndividual ,
    tcc:Simulation ;
  tcc:hasScenario :scenario-2 ;
  tcc:id "simulation-2" ;
  tcc:name "MAS to MAS Simulation"^^xsd:string ;
  tcc:descrition "The simulation demonstrates the Multi-Agent System to
    Multi-Agent System interoperability."^^xsd:string ;
  tcc:created "2019-07-25T16:26:24"^^xsd:dateTime ;
  tcc:modified "2019-07-25T16:26:24"^^xsd:dateTime ;
  .

:scenario-2
  rdf:type
    owl:NamedIndividual ,
    tcc:Scenario ;
  tcc:hasStep
    :step-1-scen-2 ,
    :step-2-scen-2 ,
    :step-3-scen-2 ;
  tcc:numberOfSteps "4"^^xsd:unsignedInt ;
  tcc:id "scenario-2" ;
  tcc:name "MAS to MAS Simulation"^^xsd:string ;
  tcc:descrition "The simulation demonstrates the Multi-Agent System to
    Multi-Agent System interoperability, namely between MASCEM and AiD-EM
    ."^^xsd:string ;
  tcc:created "2019-07-25T16:26:24"^^xsd:dateTime ;
  tcc:modified "2019-07-25T16:26:24"^^xsd:dateTime ;
  .
```

Listing 4.14: Base de Conhecimento com a configuração geral do cenário 2 (Turtle)

O primeiro StepA vai conter na sua BCS a informação apresentada no excerto de código 4.15. A fase de execução `:step-1-scen-2` vai iniciar o simulador ALBidS (linha 5). Para este fim, é necessário proceder à importação do ficheiro que contém os dados de entrada da ferramenta (linha 14), que tem como extensão esperada um `.xslx`. Os dados importados por este ficheiro são serializados em formato `string`, segundo um modelo pré-definido esperado pela ferramenta.

```
:step-1-scen-2
  rdf:type
    owl:NamedIndividual ,
    tcc:Step ;
  tcc:hasService :service-run-aid-em ;
  tcc:hasNext :step-2-scen-2 ;
  tcc:step "1"^^xsd:unsignedInt ;
```

```
8    .
9
10 :service-run-aid-em
11   rdf:type
12     owl:NamedIndividual ,
13     tcc:Service ;
14   tcc:hasInputDataSource :input-file-aid-em ;
15   .
16
17 :input-file-aid-em
18   rdf:type
19     owl:NamedIndividual ,
20     tcc:FileDataSource ;
21   tcc:filePath "file:///c:/tmdei/cs/input-aid-em.xlsx"^^xsd:anyURI ;
22   tcc:fileFormat "XLSX"^^xsd:string ;
23   .
```

Listing 4.15: Base de Conhecimento com perspetiva geral da configuração da primeira fase de execução do cenário 2 (Turtle)

A fase de execução seguinte - `:step-2-scen-2` - é equivalente à primeira. No entanto, ao invés de executar o serviço para iniciar o ALBidS, será iniciado o simulador `MASCEM`, através da execução do serviço `:service-run-mascem`. Este serviço necessita da importação de dados de entrada através de uma fonte de dados `:input-file-mascem` (linha 15). Assim que é importado, os dados são desserializados para o modelo esperado pelo agente principal do simulador MASCEM.

```
1  :step-2-scen-2
2    rdf:type
3      owl:NamedIndividual ,
4      tcc:Step ;
5    tcc:hasService :service-run-mascem ;
6    tcc:hasPrevious :step-1-scen-2 ;
7    tcc:hasNext :step-3-scen-2 ;
8    tcc:step "2"^^xsd:unsignedInt ;
9    .
10
11 :service-run-mascem
12   rdf:type
13     owl:NamedIndividual ,
14     tcc:Service ;
15   tcc:hasInputDataSource :input-file-mascem ;
16   .
17
18 :input-file-mascem
19   rdf:type
20     owl:NamedIndividual ,
21     tcc:FileDataSource ;
22   tcc:filePath "file:///c:/tmdei/cs/input-mascem.xlsx"^^xsd:anyURI ;
23   tcc:fileFormat "XLSX"^^xsd:string ;
24   .
```

Listing 4.16: Base de Conhecimento com perspetiva geral da configuração da segunda fase de execução do cenário 2 (Turtle)

No fim, é executada a terceira fase (`:step-3-scen-2`), que tem como meta dar início à execução do simulador MASCEM, com a chamada do serviço `:service-run-mascem-simumation`, apresentado pelo excerto de código 4.17. Sendo que os dados entrada já

foram introduzidos através da fase anterior, este serviço apenas contém um modelo de dados de saída, que é definido pela ontologia EMR.

```
:step-3-scen-2
  rdf:type
    owl:NamedIndividual ,
    tcc:Step ;
  tcc:hasService :service-run-mascem-simulation ;
  tcc:hasPrevious :step-2-scen-2 ;
  tcc:step "3"^^xsd:unsignedInt ;
  .

:service-run-mascem-simulation
  rdf:type
    owl:NamedIndividual ,
    tcc:Service ;
  tcc:hasOutputModel emr: ;
  .
```

Listing 4.17: Base de Conhecimento com perspetiva geral configuração da terceira fase de execução do cenário 2 (Turtle)

4.3.3 Resultados

Após a execução do SMA TOOCC, são apresentados os resultados obtidos ao utilizador. Para este problema é possível a visualização de diferentes resultados, que proporcionam uma visão do problema sob o ponto de vista das diferentes entidades envolvidas, nomeadamente: a curva de mercado para cada um dos períodos; o cruzamento entre a oferta e procura; lucro por participante; entre outros.

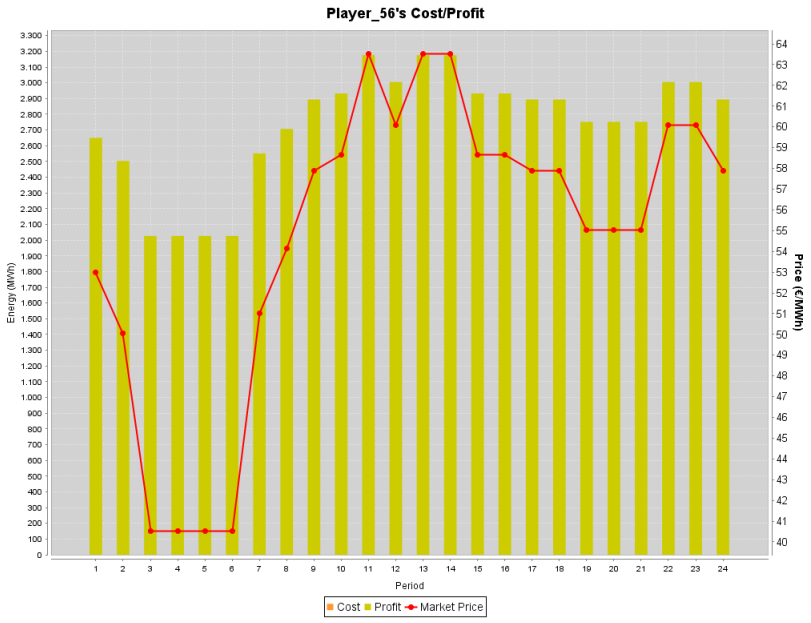

Figura 4.7: Resultado da relação entre custo e lucro para o participante vendedor de energia nº56, considerado no cenário MAS2MAS

A Figura 4.7 apresenta o resultado obtido para o participante 56, relativamente ao custo/benefício que este teve com a sua participação no mercado. Por observação à figura é possível perceber que este participante tem muitas vezes um lucro superior ao preço de mercado, havendo apenas três períodos (11, 13 e 14) onde o lucro é igual ao preço de mercado. Comparativamente ao participante 58 (presente na Figura 4.8), conseguimos perceber que este último não obteve os mesmos benefícios que o anterior.

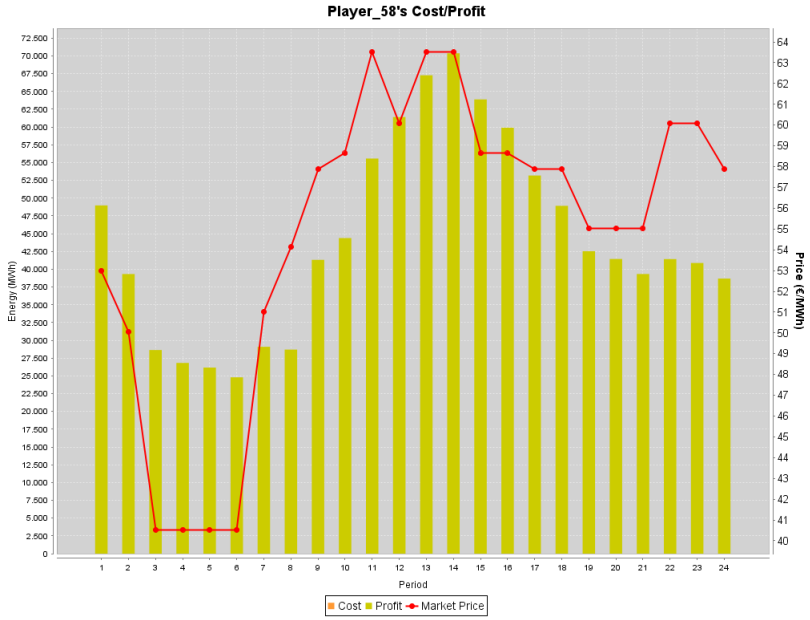

Figura 4.8: Resultado da relação entre custo e lucro para o participante vendedor de energia nº58, considerado no cenário MAS2MAS

Como análise final, para compreender se o ALBidS teve impacto positivo na participação do vendedor 56 seria necessário efetuar uma nova simulação, onde seria desconsiderada a utilização da ferramenta de suporte à decisão. Caso existam diferenças significativas, poderá concluir-se com certeza que o bom desempenho do participante deveu-se à utilização do ALBidS.

4.4 Caso de Estudo 3 - Interoperabilidade Entre Sistemas Multi-Agente e Serviços Web

O próximo caso de estudo pretende a execução de um cenário que estabeleça comunicação entre um SMA e um serviço *web*, designado por Service-to-MAS (S2MAS). Neste caso, serão considerados dois agentes que pertencem ao simulador MASGriP, nomeadamente os agentes Network Manager Agent (NM) e Facility Manager Agent (FM), que se servem de previsões efetuadas por um algoritmo disponibilizado por um serviço *web*. Ao contrário dos simuladores MASCEM e ALBidS, estes agentes não disponibilizam semântica.

4.4.1 Problema

O problema abordado neste cenário ocorre ao nível de uma micro-rede em que o objetivo principal resume-se à análise do impacto da gestão da rede no consumidor final. Além disso, o cenário trata uma visão futura da rede para o inverno do ano de 2050, estudando quais serão as necessidades e comportamentos da mesma.

Neste cenário são considerados dados relativos ao campus do ISEP-IPP, que possuem informação sobre 21 barramentos, constituídos por: 20 cargas, 20 geradores fotovoltaicos, 4 geradores eólicos, 7 sistemas de armazenamento de energia, 2 capacitores, 300 carros elétricos, e 1 fornecedor externo.

O NM tem o papel de efetuar a gestão da rede, e o FM irá representar um consumidor presente no barramento 6, que consiste no edifício N. Internamente, ambos os agentes possuem um algoritmo de escalonamento, embora com propósitos e características diferentes. O NM efetua um escalonamento da rede considerando todos os seus intervenientes, enquanto que o FM gere a energia estabelecida pelo NM pelos seus dispositivos (máquina de lavar, máquina café, sistemas de climatização, entre outros). No entanto, para o NM ser capaz de executar o seu escalonamento, este terá de receber a previsão do consumo e produção de todos os nós da rede.

4.4.2 Execução

A configuração do cenário obedece à estrutura apresentada na Figura 4.9. Sob a perspetiva de execução da ferramenta TOOCC, este cenário distingue-se dos restantes apresentados até ao momento por estabelecer uma comunicação de SMA com serviços *web*, e também pelo facto de, para a simulação do serviço na segunda fase de execução, necessitar de duas precedências ao invés de apenas uma.

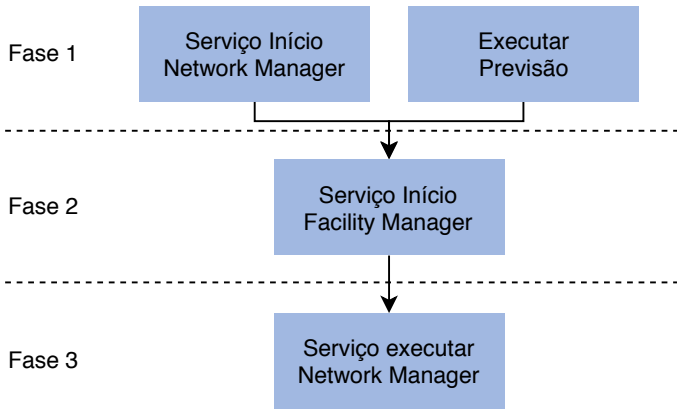

Figura 4.9: Perspetiva geral da execução do terceiro caso de estudo - S2MAS - pela ferramenta TOOCC

Para o seu funcionamento, apenas é necessário que a BCS conheça a ontologia `TCC`. O excerto de código 4.18 permite analisar a construção do terceiro cenário. Este, à semelhança dos anteriores, também é constituído por uma única simulação no `tcc:SimulationSet`. A simulação tem também apenas um cenário cujo nome atende por `:scenario-3`.

```
:simulation-set
  rdf:type
    owl:NamedIndividual ,
    tcc:SimulationSet ;
  tcc:hasSimulation
    :simulation-3 ;
  .

:simulation-3
  rdf:type
    owl:NamedIndividual ,
    tcc:Simulation ;
  tcc:hasScenario :scenario-3 ;
  tcc:id "simulation-3" ;
  tcc:name "Services to MAS Simulation"^^xsd:string ;
  tcc:descrition "The simulation demonstrates the interoperability
    between two MASGriP agents (Network Manager and Facility Manager)
    while defining Web Services to solve distinct agents' operations."^^
    xsd:string ;
  tcc:created "2019-07-25T16:26:24"^^xsd:dateTime ;
  tcc:modified "2019-07-25T16:26:24"^^xsd:dateTime ;
  .

:scenario-3
  rdf:type
    owl:NamedIndividual ,
    tcc:Scenario ;
  tcc:hasStep
    :step-1-scen-3 ,
    :step-2-scen-3 ,
    :step-3-scen-3 ;
  tcc:numberOfSteps "3"^^xsd:unsignedInt ;
  tcc:id "scenario-3" ;
   tcc:name "Services to MAS Simulation"^^xsd:string ;
  tcc:descrition "The simulation demonstrates the interoperability
    between two MASGriP agents (Network Manager and Facility Manager)
    while defining Web Services to solve distinct agents' operations."^^
    xsd:string ;
  tcc:created "2019-07-25T16:26:24"^^xsd:dateTime ;
  tcc:modified "2019-07-25T16:26:24"^^xsd:dateTime ;
  .
```

Listing 4.18: Base de Conhecimento com a configuração geral do cenário 3 (Turtle)

O StepA para a fase `:step-1-scen-3` necessita de criar dois ServAs. Tal como é apresentado no excerto de código 4.19, um deles tem como tarefa a execução do serviço `:service-start-nm`, para a iniciação do NM, enquanto que o outro tem o propósito de efetuar a previsão dos consumos dos intervenientes da rede, por meio da execução do serviço `:service-forecast-algorithm`. Estes dois serviços são precedências para execução do serviço `:service-run-fm`, que processa durante a fase de execução seguinte. Os dois serviços também recebem como dados de entrada uma estrutura em formato `JSON`.

```
:step-1-scen-3
  rdf:type
    owl:NamedIndividual ,
    tcc:Step ;
  tcc:hasService
```

```
    :service-start-nm ,
    :service-forecast-algorithm ;
    tcc:hasNext :step-2-scen-3 ;
    tcc:step "1"^^xsd:unsignedInt ;
    .

:service-forecast-algorithm
    rdf:type
    owl:NamedIndividual .
    tcc: Service ;
    tcc:hasInputDataSource :input-file-forecast-algorithm ;
    tcc:hasOutputModel : OutputForecastAlgorithm;
    tcc:hasNext :service-run-fm;
    .

:service-start-nm
    rdf:type
      owl:NamedIndividual ,
      tcc:Service ;
    tcc:hasInputDataSource :input-file-nm ;
    tcc:hasNext :service-run-fm;
    .

:input-file-forecast-algorithm
    rdf:type
      owl:NamedIndividual ,
      tcc:FileDataSource ;
    tcc:filePath "file:///c:/tmdei/cs/input-forecast.json"^^xsd:anyURI ;
    tcc:fileFormat "JSON"^^xsd:string ;
    .

:input-file-nm
    rdf:type
      owl:NamedIndividual ,
      tcc:FileDataSource ;
    tcc:filePath "file:///c:/tmdei/cs/input-nm.json"^^xsd:anyURI ;
    tcc:fileFormat "JSON"^^xsd:string ;
    .
```

Listing 4.19: Base de Conhecimento com perspetiva geral da configuração da primeira fase de execução do cenário 3 (Turtle)

O modelo dos resultados do `:service-forecast-algorithm` é demonstrado no excerto de código 4.20. Através deste é possível observar que os dados de saída são definidos através da matriz `:FacilitiesForecast`, onde cada vetor é uma entidade da rede (ou *facility*), e os itens de cada vetor são o resultado da previsão para cada período do dia.

```
:OutputForecastAlgorithm
    rdf:type owl:Class ;
    rdfs:subClassOf
      tcc:OutputModel ,
      [
        rdf:type owl:Restriction ;
        owl:onProperty tcc:hasOutputModel ;
        owl:cardinality "1"^^xsd:nonNegativeInteger ;
        owl:onClass :FacilitiesForecast
      ] ;
    .

```

```
:FacilitiesForecast
  rdf:type owl:Class ;
  rdfs:subClassOf mat:Matrix ;
  .
```

Listing 4.20: Base de Conhecimento com perspetiva geral da configuração da segunda fase de execução do cenário 3 (Turtle)

```
:step-2-scen-3
  rdf:type
    owl:NamedIndividual ,
    tcc:Step ;
  tcc:hasService :service-run-fm ;
  tcc:hasPrevious :step-1-scen-3 ;
  tcc:hasNext :step-3-scen-3 ;
  tcc:step "2"^^xsd:unsignedInt ;
  .

:service-run-fm
  rdf:type
    owl:NamedIndividual ,
    tcc:Service ;
  tcc:hasInputDataSource :input-file-fm ;
  tcc:hasInput :InputRunFMService ;
  tcc:hasPrevious :service-forecast-algorithm ;
  tcc:hasPrevious :service-start-nm ;
  .

:input-file-fm
  rdf:type
    owl:NamedIndividual ,
    tcc:FileDataSource ;
  tcc:filePath "file:///c:/tmdei/cs/input-fm.json"^^xsd:anyURI ;
  tcc:fileFormat "JSON"^^xsd:string ;
  .
:InputRunFMService
  rdf:type owl:Class ;
  rdfs:subClassOf
    tcc:InputModel ,
  [
      rdf:type owl:Restriction ;
      owl:onProperty tcc:hasInputDataSource ;
      owl:cardinality "1"^^xsd:nonNegativeInteger ;
      owl:onClass :FacilitiesForecast
    ] ;
  .
```

Listing 4.21: Base de Conhecimento com perspetiva geral da configuração da segunda fase de execução do cenário 3 (Turtle)

A segunda fase de execução denomina-se `:step-2-scen-3` e está representada no excerto de código 4.21. Este `tcc:Step` tem na sua formação apenas a execução do serviço `:service-run-fm`. Este serviço necessita que primeiramente executem os dois serviços da fase anterior devido a: (i) quando o FM inicia este vai registar-se na plataforma do NM e, portanto, precisa que este já esteja em estado pronto a receber a sua comunicação; e (ii) quando é feito o registo do FM no NM, este deve indicar qual é a sua previsão de consumo/-produção, para que quando o NM efetuar o escalonamento, este possa garantir a satisfação da demanda de energia. O serviço `:service-run-fm` recebe como modelo de entrada a

informação extraída de um ficheiro JSON, com os dados dos restantes nós da rede, além dos resultados do serviço de previsão executados anteriormente.

A terceira fase consiste em solicitar ao NM para dar início à simulação. Todo o processo de interação entre o NM com o FM é garantido por eles, sendo apenas necessário aguardar que estes terminem a sua execução. Como dados de saída, o serviço obtém: o escalonamento do NM efetuado para o dia seguinte, hora seguinte e em tempo real; cortes efetuados por consumidor; consumo total; e consumo de cada dispositivo do FM após aplicar as restrições impostas pelo NM.

```
:step-3-scen-3
  rdf:type
    owl:NamedIndividual ,
    tcc:Step ;
  tcc:hasService :service-run-nm-simulation ;
  tcc:hasPrevious :step-2-scen-3 ;
  tcc:hasOutput :OutputNMSimulation ;
  tcc:step "3"^^xsd:unsignedInt ;
  .

:service-run-nm-simulation
  rdf:type
    owl:NamedIndividual ,
    tcc:Service ;
  .
```

Listing 4.22: Base de Conhecimento com perspetiva geral da terceira da primeira fase de execução do cenário 3 (Turtle)

4.4.3 Resultados

Os resultados descritos a seguir correspondem à informação que é disponibilizada ao utilizador quando o SMA termina a simulação. A Figura 4.10 demonstra que grande parte da energia utilizada pela rede advém de fontes de produção distribuída, excluindo as primeiras horas do dia. Neste período do dia, para satisfazer as necessidades da rede é necessário comprar energia no MEE, ou utilizar a energia armazenada em baterias. Por outro lado, quando há produção de energia excedente, é possível vende-la a elementos exteriores à rede, como se pode verificar entre as 3h e as 10h.

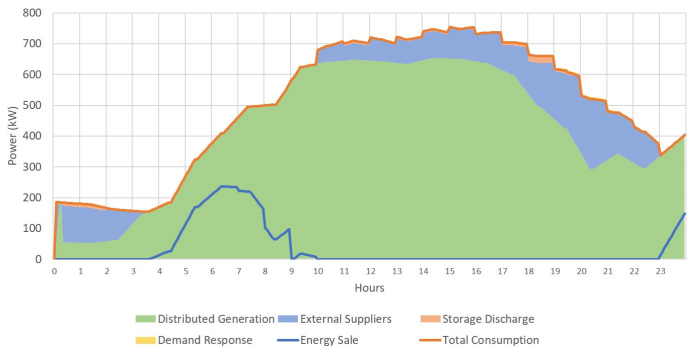

Figura 4.10: Resultado da produção de energia segundo a perspetiva do agregador, considerando o cenário S2MAS

A Figura 4.11 apresenta o escalonamento do consumo de energia da micro-rede, para cada período. Através da observação da área do consumo, é possível perceber que é realizada uma deslocação do consumo das horas de pico para períodos em que o preço da energia seja mais baixo (durante o período da tarde), pois o perfil de consumo energético é mais uniforme ao longo do dia, contrariando os tradicionais picos nos períodos da manhã e da noite. Ao deslocar o consumo são consideradas as prioridades do FM e dos restantes consumidores da rede, cujos dispositivos que são caracterizados como tendo uma prioridade mais elevada não são afetados pelo corte. Sendo que o cenário refere-se à estação do inverno, os dispositivos com maior prioridade são aqueles que ajudam a combater o frio.

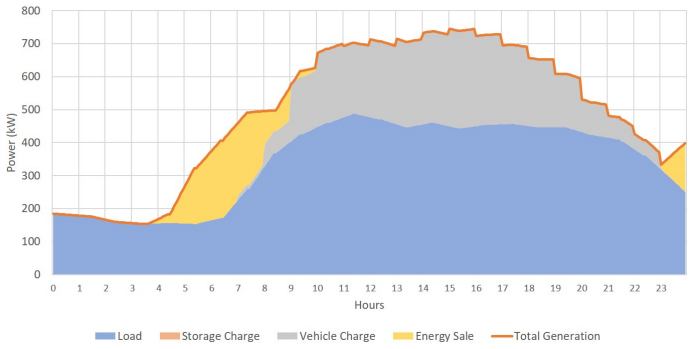

Figura 4.11: Resultado do escalonamento do consumo de energia da rede, considerando o cenário S2MAS

A Figura 4.12 apresenta o efeito que os cortes feitos pelo NM ao nível do FM. O gráfico demonstra que especialmente no período da noite, nomeadamente entre as 19h e 22h, são feitos os maiores cortes. Este facto acontece porque é neste período que ocorre grande parte do consumo da rede. Com o intuito de balancear o consumo na mesma, o NM solicita ao FM para que este desloque os consumos para horas com menos afluência, como por exemplo durante o período da tarde ou após as 23h.

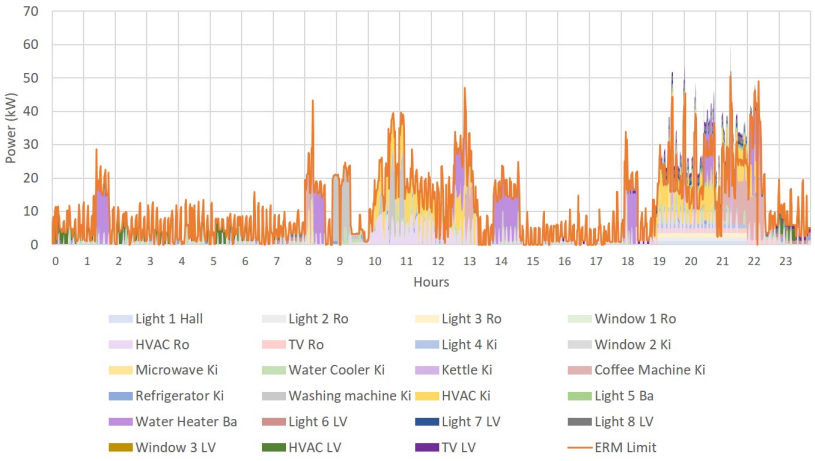

Figura 4.12: Resultado da aplicação dos cortes de energia ao nível do FM, considerando o cenário S2MAS

4.5 Caso de Estudo 4 - Comparação de Resultados

O quarto e último caso de estudo tem como objetivo demonstrar a comparação de resultados, com o intuito de apontar a utilidade desta funcionalidade para a análise de cenários. Este estudo pretende comparar diferentes algoritmos de previsão para consumos de energia de um edifício.

4.5.1 Problema

O problema deste caso de estudo pretende apurar qual é o melhor método de previsão para antever o consumo de tomadas para o dia seguinte. Os dados são extraídos do repositório de dados do GECAD, onde são armazenados dados de consumo e produção, em tempo real.

Serão considerados cinco algoritmos de previsão distintos, nomeadamente um algoritmo de previsão Neural Artificial Network (ANN) , um algoritmo baseado em Support Vector Machine (SVM), um algoritmo baseado em Hybrid Neural Fuzzy Inference System (HyFIS), um algoritmo que aplica Wang and Mendel's technique (WM), e por fim a metodologia MOGUL.

4.5.2 Execução

Contrariamente aos casos de estudo anteriores, este estudo apenas possui uma única fase de execução, como se pode observar pela Figura 4.13.

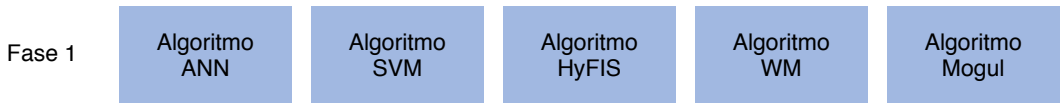

Figura 4.13: Perspetiva geral da execução do quarto caso de estudo - comparação de resultados - pela ferramenta TOOCC

O excerto de código 4.23 apresenta a perspetiva geral da configuração para o atual caso de estudo. Nele, é possível observar que a simulação é constituída pelo cenário `:scenario-4`, que tem apenas uma única fase de execução `:step-1-scen-4` (linha 26).

```
1  :simulation-set
2    rdf:type
3      owl:NamedIndividual ,
4      tcc:SimulationSet ;
5    tcc:hasSimulation
6      :simulation-4;
7    .
8
9  :simulation-4
10   rdf:type
11     owl:NamedIndividual ,
12     tcc:Simulation ;
13   tcc:hasScenario :scenario-4 ;
14   tcc:id "simulation-4" ;
15   tcc:name "Forecast Algorithms Comparison"^^xsd:string ;
```

```
  tcc:description "The simulation demonstrates the comparison amoung
    different forecast algorithms."^^xsd:string ;
  tcc:created "2019-07-25T16:26:24"^^xsd:dateTime ;
  tcc:modified "2019-07-25T16:26:24"^^xsd:dateTime ;
  .

:scenario-4
  rdf:type
    owl:NamedIndividual ,
    tcc:Scenario ;
  tcc:hasStep
    :step-1-scen-4 ;
  tcc:numberOfSteps "1"^^xsd:unsignedInt ;
  tcc:id "scenario-4" ;
  tcc:name "Forecast Algorithms Comparison"^^xsd:string ;
  tcc:description "The simulation demonstrates the comparison amoung
    different forecast algorithms."^^xsd:string ;
  tcc:created "2019-07-25T16:26:24"^^xsd:dateTime ;
  tcc:modified "2019-07-25T16:26:24"^^xsd:dateTime ;
  .
```

Listing 4.23: Base de Conhecimento com perspetiva geral da configuração do cenário 4 (Turtle)

Sendo que a fase `:step-1-scen-4` tem na sua configuração a definição de cinco serviços, o StepA vai proceder à criação de também cinco ServA, que vão executar em paralelo. Adicionalmente, é considerada a propriedade `tcc:hasComparable`, que vai permitir ao TMA identificar que o utilizador pretende comparar diretamente os valores obtidos pela execução.

```
:step-1-scen-4
  rdf:type
    owl:NamedIndividual ,
    tcc:Step ;
  tcc:hasService :service-forecast-ann ,
  service-forecast-svm ,
  service-forecast-hyfis ,
  service-forecast-wm ,
  service-forecast-mogul ;
  tcc:step "1"^^xsd:unsignedInt ;
  .

:service-forecast-ann
  rdf:type
    owl:NamedIndividual ,
    tcc:Service ;
  tcc:hasInputDataSource :input-file-forecast-ann ;
  tcc:hasComparable :service-forecast-svm , service-forecast-hyfis ,
    service-forecast-wm, service-forecast-mongo ;
  .

:service-forecast-svm
  rdf:type
    owl:NamedIndividual ,
    tcc:Service ;
  tcc:hasInputDataSource :input-file-forecast-svm ;
  tcc:hasComparable :service-forecast-ann , service-forecast-hyfis ,
    service-forecast-wm, service-forecast-mongo ;
  .

```

```
:service-forecast-hyfis
  rdf:type
    owl:NamedIndividual ,
    tcc:Service ;
  tcc:hasInputDataSource :input-file-forecast-hyfis ;
  tcc:hasComparable :service-forecast-ann , service-forecast-svm ,
    service-forecast-wm, service-forecast-mongo ;
  .

(...)

:input-file-forecast-ann
  rdf:type
    owl:NamedIndividual ,
    tcc:FileDataSource ;
  tcc:filePath "file:///c:/tmdei/cs/input-ann.xlsx"^^xsd:anyURI ;
  tcc:fileFormat "XLSX"^^xsd:string ;
  .

(...)
```

Listing 4.24: Adaptação da Base de Conhecimento da configuração da primeira fase de execução do cenário 4 (Turtle)

4.5.3 Resultados

Após a execução do SMA TOOCC são apresentados os resultados ao utilizador. Desta vez, como foram executados cinco serviços, é elaborado um gráfico com os resultados, permitindo ao utilizador estabelecer a comparação entre os mesmos, e compreender qual é aquele que se adequa melhor ao seu problema.

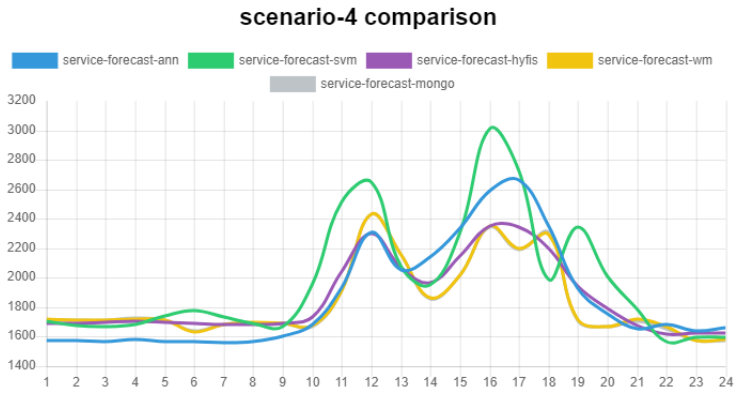

Figura 4.14: Perspetiva geral da execução do quarto caso de estudo - comparação de resultados - pela ferramenta TOOCC

A disponibilização de uma funcionalidade para comparar resultados pode trazer grandes benefícios para o utilizador, nomeadamente na compreensão de qual é o impacto da configuração dos serviços no resultado final, ou até mesmo avaliar o desempenho dos mesmos na aplicação de um problema.

4.6 Considerações Finais

Neste capítulo foram apresentados quatro casos de estudo que demonstram a flexibilidade da ferramenta TOOCC na construção e simulação de cenários aplicados a diferentes contextos e problemas.

No primeiro caso de estudo é mostrada a construção de um cenário que permite diferentes serviços *web* comunicarem. Com esta finalidade, foi abordado um problema em que é aplicado um programa de DR e é pretendido efetuar uma remuneração justa aos consumidores, por parte do agregador. Neste contexto são considerados três algoritmos, que executam em fases diferentes, garantindo que os resultados de um serviço são utilizados pelo serviço seguinte, e assim sucessivamente. As fases são executadas pela seguinte ordem: escalonamento da energia, que resulta no valor de energia reduzido por cada consumidor; a agregação dos consumidores em grupos, considerando a quantidade de energia reduzida; e, perante os grupos formados na fase anterior, apurar qual é o melhor método de remuneração, em que se pretende que o valor seja suficiente baixo para trazer vantagens ao agregador e, em simultâneo, seja aliciante para o consumidor continuar a participar nos programas de DR.

Para o segundo caso de estudo considerou-se um cenário em que a interoperabilidade é feita com dois SMA, nomeadamente o simulador MASCEM e o simulador ALBidS. O MASCEM faz a simulação da obtenção do preço de mercado, enquanto que o ALBidS permite fornecer apoio à decisão aos participantes no MEE. Neste estudo é considerado o caso particular do 56º participante, que corresponde a um vendedor. Estes dois simuladores possuem inteligência e conhecimento necessário para conversarem entre si, portanto, a execução do cenário resulta na iniciação das plataformas e garantir que estas recebem a informação necessária para a sua execução.

No terceiro caso de estudo, o objetivo consiste na demonstração de um cenário que seja capaz de interligar um serviço *web* com um SMA. Assim, o problema considerado é o estudo do impacto da aplicação do escalonamento de uma micro-rede por parte do gestor da rede, no consumidor final, que terá de ajustar o seu perfil de consumo de acordo com as restrições do gestor da rede. Neste caso, o gestor da rede é representado pelo agente NM, que pertence ao simulador MASGriP, enquanto que o consumidor final corresponde ao agente FM do mesmo sistema. De forma idêntica ao segundo caso de estudo, os agentes considerados neste cenário apenas precisam ser iniciados, e depois disso são capazes de efetuar a simulação entre si. Contudo, para que o NM possa desempenhar a sua tarefa terá de lhe ser fornecida a previsão do consumo da rede, para que este possa então fazer o escalonamento energético apropriado.

Por fim, o último caso de estudo demonstra com um pequeno exemplo como é possível tirar partido daTOOCC para a comparação entre diferentes resultados. Este estudo é baseado na comparação de cinco algoritmos de previsão (ANN, SVM, HyFIS, WM e MOGUL) que apresentam resultados muito distintos para os dados utilizados.

Pode concluir-se que a ferramenta TOOCC proporciona a comunicação e integração de uma grande diversidade de ferramentas de simulação, algoritmos e informação, que permite a conceção de cenários que abordam problemas relativos a qualquer domínio dos SEE.

Capítulo 5

Conclusões

5.1 Síntese e Conclusões

A implementação em grande escada das fontes de geração distribuída de energia, assim como as metas impostas pela União Europeia perante o novo paradigma climático, provocam severas alterações no setor, que se vem a transformar continuamente para dar resposta a estes (e outros) desafios. Contudo, sendo este um ambiente pautado por um elevado grau de complexidade, flexibilidade e dinamismo, são várias as ferramentas existentes na literatura que estudam o seu comportamento e modelos, procurando superar os desafios encontrados. Por outro lado, embora possa ser encontrado um grande número de ferramentas, o domínio de aplicação das mesmas é quase sempre restrito, com a orientação à solução de um problema particular.

Assim, com o intuito de retirar vantagem do facto de já se encontrarem diversas ferramentas disponíveis para o estudo dos sistemas de energia elétrica, identificou-se a necessidade de combinar as capacidades individuais das mesmas, de modo a que estas se comportem como uma só entidade. Deste modo, torna-se possível estudar problemas que intersetam vários domínios do setor, refletindo a sua realidade. Todavia, a co-simulação entre ferramentas traz diversos desafios. Possibilitar a interoperabilidade entre sistemas heterogéneos, assim como criar cenários que possam abranger diversos domínios, requerem uma grande flexibilidade, especialmente se forem consideradas várias ferramentas, onde as suas configurações, e quantidade de dados, exigem muita atenção do utilizador.

Esta dissertação apresenta o conceito, desenvolvimento e implementação da solução TOOCC - Tools Control Center - como um sistema que permite a interoperabilidade entre ferramentas de estudo e simulação heterogéneas, no âmbito dos SEE. Esta solução tem a capacidade de construir, simular e analisar cenários relativos problemas complexos, que integram múltiplos simuladores e serviços *web*, para o estudo do comportamento das entidades envolvidas, novos modelos de mercado, perfis de consumo, gestão energética, entre outros conceitos e relações.

A ferramenta TOOCC é constituída por diversos módulos que cooperam entre si para o objetivo do sistema, nomeadamente uma aplicação *web* que faz a interface entre o sistema e o utilizador, permitindo-lhe criar e configurar os cenários, além de, posteriormente, analisar os resultados obtidos pela simulação; ontologias que permitem a definição de vocabulário para os serviços "conversarem"entre si; um SMA que é capaz de interpretar e usar as ontologias durante o processo de simulação; uma API para efetuar a ponte entre a plataforma *web* e o SMA; e outros módulos com menor dimensão, que fornecem métodos auxiliares aos restantes módulos.

O módulo da interface gráfica do utilizador permite com que este acompanhe todo o processo de preparação e simulação do cenário. Através dela são disponibilizados modelos e dados que advêm de diferentes repositórios, de forma a que o utilizador os possa integrar nas suas simulações a fim de as enriquecer. No processo de configuração de cenário(s) o utilizador pode escolher quais resultados pretende comparar para que a informação já lhe seja apresentada diretamente na fase de analise dos resultados, juntamente com os restantes resultados de cenário(s) de simulação.

Como metodologia para permitir interoperabilidade entre os diferentes serviços, o TOOCC age como intermediário nas comunicações entre os sistemas, recorrendo ao uso de ontologias. No âmbito deste trabalho são utilizados dois tipos de ontologias. O primeiro tipo são as ontologias de domínio utilizadas pelas aplicações/serviços com os quais a TOOCC comunica. Estas ontologias foram disponibilizadas publicamente com o objetivo de promover a interligação com outras ferramentas e reutilização por terceiros. O outro tipo é uma ontologia aplicacional, desenvolvida especialmente para auxiliar a TOOCC no processo de simulação por meio da configuração das simulações. Recorrendo a esta ontologia o SMA tem a capacidade de executar os cenários de forma mais dinâmica, flexível e inteligente.

O SMA é utilizado para executar o processo de simulação através da base de conhecimento semântica gerada pela configuração do cenário. Com a utilização desta, o sistema gera todos os agentes necessários e comportamentos a adotar, nomeadamente dependências de execução entre ferramentas. A utilização de um SMA como opção para executar as simulações tem diversas vantagens. Através da utilização de agentes é possível a decomposição do problema, separação de tarefas, distribuir os agentes por máquinas pré-configuradas, garantir a escabilidade do sistema, entre outros.

Módulos complementares com menor dimensão foram desenvolvidos para auxiliar no processo de simulação, particularmente na interpretação das ontologias, mapeamentos validações e transformações de dados.

Para demonstrar o trabalho desenvolvido no âmbito desta dissertação, foram apresentados quatro casos de estudo que permitem demonstrar o funcionamento da ferramenta na sua principal função, que consiste na interoperabilidade entre ferramentas. Estes casos de estudo recorrem a diferentes serviços e possuem diferentes propósitos, com vista em demonstrar a versatilidade na utilização da TOOCC.

O primeiro caso de estudo pretende demonstrar como é estabelecida a interoperabilidade entre serviços web, recorrendo à aplicação de um problema relacionado com programas *DR*, designadamente com a remuneração do consumidor. Durante o caso de estudo são considerados três serviços que executam de forma encadeada: (i) algoritmo de escalonamento de energia que indica a redução que o utilizador terá de efetuar; (ii) algoritmo de agregação de consumidores em grupos, considerando a quantidade de energia reduzida; e (iii) algoritmo de remuneração que permite estudar qual será o melhor método a considerar para remuneração pelo corte de energia. Durante o estudo é possível compreender como este é construído, interpretado pelos agentes e como o utilizador pode interpretar os resultados.

O segundo caso de estudo apresenta como é feita a co-simulação entre dois SMA, que possuem inteligência e são capazes de agir autonomamente. Ao contrário do caso anterior em que a lógica é desenhada através do encadeamento dos serviços, com a utilização dos SMA apenas é necessário provê-los da informação que eles necessitam de conhecer para iniciar a sua execução, e com isso eles são capazes de comunicar diretamente entre si. Para a demonstração utilizou-se os simuladores MASCEM e ALBidS. Sendo que estes simuladores

atuam na área dos MEE, o problema considerado consiste em efetuar uma simulação de mercado recorrendo ao MASCEM, e fornecer apoio à decisão a um dos seus participantes através da utilização do ALBidS.

O terceiro caso de estudo tem como objetivo a interoperabilidade entre serviços *web* e SMA. Para isso, utilizou-se um serviço de previsão que fornece dados necessários para o funcionamento do SMA MASGriP, mais especificamente dois agentes que fazem parte desse sistema: Facility Manager (FM) e Network Manager (NM). Este caso difere dos restantes, pois apesar da dinâmica NM e FM ser autónoma, analogamente ao MASCEM e ALBidS, estes agentes são integrados com serviços que não utilizam na sua essência. O problema abordado neste caso de estudo abarca a análise do impacto da gestão do escalonamento de energia por parte de um gestor da rede, ao nível do consumidor.

Por fim, o quarto caso de estudo demonstra como pode ser utilizada a comparação de resultados em benefício do utilizador. Para tal, considerou-se cinco métodos de previsão distintos (ANN, SVM, HyFIS, WM e MOGUL), que executam simultaneamente e que, no final, são comparados os seus resultados a fim do utilizador compreender qual deles tem mais valor para o seu objetivo.

A *framework* TOOCC vem colmatar uma necessidade identificada na literatura atual, onde além de existir um número reduzido de soluções que permitam a co-simulação entre sistemas, estas não possuem a flexibilidade necessária para a construção de cenários, além de continuarem a operar em domínios restritos. Adicionalmente, os resultados desta dissertação demonstram que a TOOCC simplifica o processo de configuração de cenários com um elevado grau de complexidade, e a sua execução é feita de forma automática, sem intervenção humana, e sem necessidade de reprogramar o sistema para cada cenário. Esta solução é uma ferramenta que permite ir de encontro às novas necessidades dos SEE, e mais especificamente, do seu estudo e simulação.

Concluindo, é possível afirmar que os objetivos desta dissertação foram cumpridos na sua totalidade, e que foi provada a contribuição da ferramenta para o meio científico, através de publicações e prémio, referidos na secção introdutória deste documento.

5.2 Trabalho Futuro

Uma vez que os SEE estão em constante evolução, também se pretende que a ferramenta TOOCC possa acompanhar esses avanços, bem como melhorar a sua tecnologia, eficiência e funcionalidades. Numa perspetiva geral, espera-se a adição contínua de novos serviços à ferramenta, que permitam a solução de problemas em contextos diferentes daqueles já existentes.

Uma vez que o desenvolvimento da ferramenta está relacionado com o projeto MASSociety, ainda são esperadas muitas alterações da ferramenta. Segue-se uma lista de algumas sugestões de trabalho futuro:

- Pretende-se aumentar a base de conhecimento semântica e, a partir dela, tornar algumas funcionalidades mais flexíveis e dinâmicas. Por exemplo, gerar componentes da interface gráfica através de uma ontologia, em que estão definidos os componentes a criar e as suas restrições;

- É desejado investir em novas funcionalidades na comparação de resultados. Permitir que sejam feitas comparações entre resultados que possuem dados com características diferentes (como a granularidade) mas que possuam o mesmo tipo e que, para poderem ser comparados, necessitam de sofrer ser mapeamentos;
- Adicionar um sistema de utilizadores e permitir que os utilizadores possam guardar uma lista de simulações, assim como permitir que estas continuem a executar em *offline*;
- Considerar ciclos e condições na configuração dos cenários, potenciando simulações mais complexas, em que a cada iteração o modelo de entrada do serviço seguinte tem de ser atualizado;
- Adição de uma ferramenta para a construção de gráficos ao gosto do utilizador, em que ele pode incluir a informação que desejar;
- Inclusão de um *dashboard* para a visualização de dados armazenados nos repositórios;
- Exportação dos dados das simulações em diversos formatos (figuras, tabelas, ficheiro `csv`, entre outros.

Bibliografia

A. Koen, Peter et al. (2002). «Fuzzy Front End: Effective Methods, Tools, and Techniques». Em: *The PDMA Toolbox 1 for New Product Development*. Ed. por Stephen Somermeyer Paul Belliveau, Abbie Griffin, pp. 5–35.

Anderson, David et al. (2012). «Intelligent Design"Real-Time Simulation for Smart Grid Control and Communications Design». Em: *IEEE Power and Energy Magazine* 10.1, pp. 49–57. issn: 15407977. doi: 10.1109/MPE.2011.943205. url: http://ieeexplore.ieee.org/document/6102582/.

Anderson, Kyle e Amit Narayan (2011). «Simulating integrated volt/var control and distributed demand response using GridSpice». Em: *2011 IEEE First International Workshop on Smart Grid Modeling and Simulation (SGMS)*. IEEE, pp. 84–89. isbn: 978-1-4673-0195-4. doi: 10.1109/SGMS.2011.6089204. url: http://ieeexplore.ieee.org/document/6089204/.

Antoine, J P e M Stubbe (1992). «EUROSTAG, software for the simulation of power system dynamics. Its application to the study of a voltage collapse scenario». Em: *IEE Colloquium on Interactive Graphic Power System Analysis Programs*, pp. 5/1–5/4.

Baader, Franz et al., eds. (2003). *The Description Logic Handbook: Theory, Implementation, and Applications*. New York, NY, USA: Cambridge University Press. isbn: 0-521-78176-0.

Barkema, Harry, Joel Baum e B Mannix (2002). «Management challenges in a new time». Em: *Academy of Management Journal* 45.

Bastian, Jens et al. (2011). «Master for Co-Simulation Using FMI». Em: pp. 115–120. doi: 10.3384/ecp11063115. url: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.385.2988%7B%5C&%7Drep=rep1%7B%5C&%7Dtype=pdf%20http://www.ep.liu.se/ecp/article.asp?issue=063%7B%5C%%7D26article=14.

Bellifemine, Fabio Luigi. et al. (2007). *Developing multi-agent systems with JADE*. John Wiley, p. 286. isbn: 9780470058404. url: https://www.wiley.com/en-us/Developing+Multi+Agent+Systems+with+JADE-p-9780470058404.

Canito, Alda et al. (2019). «Semantic Web Services for Multi-Agent Systems Interoperability». Em: *Progress in Artificial Intelligence*. Ed. por Paulo Moura Oliveira, Paulo Novais e Luís Paulo Reis. Cham: Springer International Publishing, pp. 606–616. isbn: 978-3-030-30244-3.

Collis, J C et al. (1998). «The ZEUS agent building tool-kit». Em: *BT Technol J Vol* 16.3. url: https://link.springer.com/content/pdf/10.1023%7B%5C%%7D2FA%7B%5C%%7D3A1009673714049.pdf.

Cui-Mei, Bao (2009). «Combining Intelligent Agent with the Semantic Web Services for Building an e-Commerce System». Em: *2009 IEEE International Conference on e-Business Engineering*. IEEE, pp. 371–376. isbn: 978-0-7695-3842-6. doi: 10.1109/ICEBE.2009.58. url: http://ieeexplore.ieee.org/document/5342087/.

Davis, Randall (1980). «Report on the Workshop on Distributed AI». Em: url: https://pdfs.semanticscholar.org/d4f3/54538a2c0084917f37327d4d3064478dc983.pdf.

Desouza, Kevin C et al. (2009). «Crafting organizational innovation processes». Em: *Innovation: Management, Policy and Practice* 11.1, p. 6. issn: 1447-9338. doi: 10.5172/impp.453.11.1.6.

Dignum, Frank e Mark Greaves (2000). «Issues in Agent Communication: An Introduction». Em: *Issues in Agent Communication*. Ed. por Frank Dignum e Mark Greaves. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 1–16. isbn: 978-3-540-40028-8. doi: 10.1007/10722777_1. url: https://doi.org/10.1007/10722777%7B%5C_%7D1.

DIgSILENT (2018). *DIgSILENT: Power System Software & Engineering*. Acedido em fevereiro 2018. url: https://www.digsilent.de/en/.

Elamy, A (2005). «Perspectives in agent-based technology». Em: *AgentLink News* 18, pp. 19–22.

Engelmore, Robert e A. J. (Antony J.) Morgan (1988). *Blackboard systems*. Addison-Wesley, p. 602. isbn: 0201174316. url: https://dl.acm.org/citation.cfm?id=576022.

European Commission (2013). *Energy challenges and policy*. Acedido em junho 2017. url: http://ec.europa.eu/europe2020/pdf/energy2_en.pdf.

– (2017). *2020 climate & energy package*. Acedido em junho 2017. url: https://ec.europa.eu/clima/policies/strategies/2020_en.

Ferber, Jacques (2001). «Multi-Agent System: An Introduction to Distributed Artificial Intelligence». Em: *J. Artificial Societies and Social Simulation* 4.

Finin, Tim et al. (1994). «KQML As an Agent Communication Language». Em: *Proceedings of the Third International Conference on Information and Knowledge Management*. CIKM '94. Gaithersburg, Maryland, USA: ACM, pp. 456–463. isbn: 0-89791-674-3. doi: 10.1145/191246.191322. url: http://doi.acm.org/10.1145/191246.191322.

FIPA (2001a). *FIPA Brokering Interaction Protocol Specification*. Acedido em fevereiro 2018. url: http://www.fipa.org/specs/fipa00033/XC00033F.html.

– (2001b). *FIPA English Auction Interaction Protocol Specification*. Acedido em fevereiro 2018. url: http://www.fipa.org/specs/fipa00031/XC00031F.html.

– (2001c). *FIPA Propose Interaction Protocol Specification*. Acedido em fevereiro 2018. url: http://www.fipa.org/specs/fipa00036/SC00036H.html.

– (2002a). *FIPA CCL Content Language Specification*. Acedido em fevereiro 2018. url: http://www.fipa.org/specs/fipa00009/XC00009B.html.

– (2002b). *FIPA Contract Net Interaction Protocol Specification*. Acedido em fevereiro 2018. url: http://www.fipa.org/specs/fipa00029/SC00029H.html.

– (2002c). *FIPA Query Interaction Protocol Specification*. Acedido em fevereiro 2018. url: http://www.fipa.org/specs/fipa00027/SC00027H.html.

– (2002d). *FIPA Request Interaction Protocol Specification*. Acedido em fevereiro 2018. url: http://www.fipa.org/specs/fipa00026/SC00026H.html.

– (2002e). *FIPA SL Content Language Specification*. Acedido em fevereiro 2018. url: http://www.fipa.org/specs/fipa00008/SC00008I.html.

– (2004). *FIPA Agent Management Specification*. Acedido em fevereiro 2018. url: http://www.fipa.org/specs/fipa00023/SC00023K.html.

Fornara, Nicoletta e Marco Gambardella (2003). «Interaction and Communication among Autonomous Agents in Multiagent Systems». Em: url: https://doc.rero.ch/record/4379/files/1%7B%5C_%7D2003COM002.pdf.

Gamma, Erich et al. (1993). «Design Patterns: Abstraction and Reuse of Object-Oriented Design». Em: *ECOOP' 93 — Object-Oriented Programming*. Ed. por Oscar M. Nierstrasz. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 406–431. isbn: 978-3-540-47910-9.

General Electric (2018). *PSLF*. Acedido em fevereiro 2018. url: https://www.geenergyconsulting.com/practice-area/software-products/pslf.

Genesereth, M R e R E Fikes (1992). «Knowledge Interchange Format, Version 3.0 Reference Manual». Em: *Interchange* Logic-92-1, pp. 1–68. doi: 10.1.1.54.8601. url: https://www.cs.auckland.ac.nz/courses/compsci367s2c/resources/kif.pdf%20http://logic.stanford.edu/kif/Hypertext/kif-manual.html.

Georg, Hanno et al. (2012). «A HLA based simulator architecture for co-simulating ICT based power system control and protection systems». Em: *2012 IEEE Third International Conference on Smart Grid Communications (SmartGridComm)*. IEEE, pp. 264–269. isbn: 978-1-4673-0910-3. doi: 10.1109/SmartGridComm.2012.6485994. url: http://ieeexplore.ieee.org/document/6485994/.

Georgilakis, Pavlos S. e Nikos D. Hatziargyriou (2013). «Optimal distributed generation placement in power distribution networks: Models, methods, and future research». Em: *IEEE Transactions on Power Systems* 28.3, pp. 3420–3428. issn: 08858950. doi: 10.1109/TPWRS.2012.2237043.

Gruber, Thomas R (1993). «A translation approach to portable ontology specifications». Em: *Knowledge Acquisition* 5.2, pp. 199–220. issn: 1042-8143. doi: https://doi.org/10.1006/knac.1993.1008. url: http://www.sciencedirect.com/science/article/pii/S1042814383710083.

Gruber, Thomas R. (1995). «Toward principles for the design of ontologies used for knowledge sharing?» Em: *International Journal of Human-Computer Studies* 43.5-6, pp. 907–928. issn: 1071-5819. doi: 10.1006/IJHC.1995.1081. url: https://www.sciencedirect.com/science/article/pii/S1071581985710816.

Gruninger, Michael e Jintae Lee (2002). «Introduction». Em: *Communications of the ACM* 45.2, pp. 39–41. issn: 00010782. doi: 10.1145/503124.503146. url: http://portal.acm.org/citation.cfm?doid=503124.503146.

Haan, J.E.S. de et al. (2011a). «Social interaction interface for performance analysis of smart grids». Em: *2011 IEEE First International Workshop on Smart Grid Modeling and Simulation (SGMS)*. IEEE, pp. 79–83. isbn: 978-1-4673-0195-4. doi: 10.1109/SGMS.2011.6089202. url: http://ieeexplore.ieee.org/document/6089202/.

– (2011b). «Social interaction interface for performance analysis of smart grids». Em: *2011 IEEE First International Workshop on Smart Grid Modeling and Simulation (SGMS)*. IEEE, pp. 79–83. isbn: 978-1-4673-0195-4. doi: 10.1109/SGMS.2011.6089202. url: http://ieeexplore.ieee.org/document/6089202/.

Hadzic, Maja et al. (2009). «Introduction to Ontology». Em: *Ontology-Based Multi-Agent Systems*. Ed. por Janusz Kacprzyk. Berlin Heidelberg: Springer, Berlin, Heidelberg. Cap. 3, pp. 37–60. doi: 10.1007/978-3-642-01904-3_3. url: http://link.springer.com/10.1007/978-3-642-01904-3%7B%5C_%7D3.

Harp, Steven A. et al. (2000). «SEPIA. A simulator for electric power industry agents». Em: *IEEE Control Systems Magazine* 20.4, pp. 53–69. doi: 10.1109/37.856179.

Hepp, Martin (2008). «Ontologies: State of the Art, Business Potential, and Grand Challenges». Em: *Ontology Management*. Boston, MA: Springer US, pp. 3–22. doi: 10.1007/978-0-387-69900-4_1. url: http://link.springer.com/10.1007/978-0-387-69900-4%7B%5C_%7D1.

Homer (2018). *HOMER Energy*. Acedido em fevereiro 2018. url: https://www.homerenergy.com/.

Hopkinson, Kenneth et al. (2006). «EPOCHS: A platform for agent-based electric power and communication simulation built from commercial off-the-shelf components». Em:

IEEE Transactions on Power Systems 21.2, pp. 548–558. issn: 08858950. doi: 10.1109/TPWRS.2006.873129. url: http://ieeexplore.ieee.org/document/1626358/.

Howell, Shaun et al. (2017). «Towards the next generation of smart grids: Semantic and holonic multi-agent management of distributed energy resources». Em: *Renewable and Sustainable Energy Reviews* 77, pp. 193–214. issn: 13640321. doi: 10.1016/j.rser.2017.03.107. url: http://linkinghub.elsevier.com/retrieve/pii/S1364032117304392.

Huang, Chun Che et al. (2010). «The agent-based negotiation process for B2C e-commerce». Em: *Expert Systems with Applications* 37.1, pp. 348–359. issn: 09574174. doi: 10.1016/j.eswa.2009.05.065. url: https://www.sciencedirect.com/science/article/pii/S0957417409004990.

Huhns, Michael N e Larry M Stephens (1999). «Multiagent Systems». Em: ed. por Gerhard Weiss. Cambridge, MA, USA: MIT Press. Cap. Multiagent, pp. 79–120. isbn: 0-262-23203-0. url: http://dl.acm.org/citation.cfm?id=305606.305608.

Jain, Sanjay et al. (2017). «Distributed generation deployment: State-of-the-art of distribution system planning in sustainable era». Em: *Renewable and Sustainable Energy Reviews* 77, pp. 363–385. issn: 1364-0321. doi: https://doi.org/10.1016/j.rser.2017.04.024. url: http://www.sciencedirect.com/science/article/pii/S1364032117305245.

Jennings, Nicholas R., Katia Sycara e Michael Wooldridge (1998). «A Roadmap of Agent Research and Development». Em: *Autonomous Agents and Multi-Agent Systems* 1.1, pp. 7–38. issn: 13872532. doi: 10.1023/A:1010090405266. url: http://link.springer.com/10.1023/A:1010090405266.

Knieps, Günter (2013). «Renewable Energy, Efficient Electricity Networks, and Sector-Specific Market Power Regulation». Em: *Evolution of Global Electricity Markets: New Paradigms, New Challenges, New Approaches*. Ed. por Fereidoon P Sioshansi. Boston: Academic Press, pp. 147–168. isbn: 9780123978912. doi: 10.1016/B978-0-12-397891-2.00006-7.

Li, W. et al. (2011). «VPNET: A co-simulation framework for analyzing communication channel effects on power systems». Em: *2011 IEEE Electric Ship Technologies Symposium*. IEEE, pp. 143–149. isbn: 978-1-4244-9272-5. doi: 10.1109/ESTS.2011.5770857. url: http://ieeexplore.ieee.org/document/5770857/.

Lin, Hua, Santhoshkumar Sambamoorthy et al. (2011). «Power system and communication network co-simulation for smart grid applications». Em: *IEEE PES Innovative Smart Grid Technologies Conference Europe, ISGT Europe*. IEEE, pp. 1–6. isbn: 9781612842189. doi: 10.1109/ISGT.2011.5759166. url: http://ieeexplore.ieee.org/document/5759166/.

Lin, Hua, Santhosh S. Veda et al. (2012). «GECO: Global Event-Driven Co-Simulation Framework for Interconnected Power System and Communication Network». Em: *IEEE Transactions on Smart Grid* 3.3, pp. 1444–1456. issn: 1949-3053. doi: 10.1109/TSG.2012.2191805. url: http://ieeexplore.ieee.org/document/6200399/.

Lindgreen, Adam e Finn Wynstra (2005). «Value in business markets: What do we know? Where are we going?» Em: *Industrial Marketing Management* 34.7, pp. 732–748. issn: 0019-8501. doi: 10.1016/J.INDMARMAN.2005.01.001. url: https://www.sciencedirect.com/science/article/pii/S0019850105000027.

Luck, Michael e Mark D'Inverno (1995). «A Formal Framework for Agency and Autonomy». Em: *In Proceedings of the First International Conference on Multi-Agent Systems*. AAAI Press / MIT Press, pp. 254–260.

Martin, Dl, Aj Cheyer e Db Moran (1999). «The Open Agent Architecture: A Framework for Building Distributed Software Systems». Em: *Applied Artificial Intelligence* 13, pp. 91–

128. issn: 0883-9514. doi: 10.1080/088395199117504. url: http://www.ai.sri.com/%7B~%7Dcheyer/papers/oaa.pdf%20http://www.tandfonline.com/doi/abs/10.1080/088395199117504.

Marzougui, Borhen e Kamel Barkaoui (2013). «Interaction Protocols in Multi-Agent Systems based on Agent Petri Nets Model». Em: 4.

McArthur, Stephen D. J. et al. (2007a). «Multi-Agent Systems for Power Engineering Applications—Part II: Technologies, Standards, and Tools for Building Multi-agent Systems». Em: *IEEE Transactions on Power Systems* 22.4, pp. 1753–1759. issn: 0885-8950. doi: 10.1109/TPWRS.2007.908472. url: http://ieeexplore.ieee.org/document/4349107/.

– (2007b). «Multi-Agent Systems for Power Engineering Applications—Part II: Technologies, Standards, and Tools for Building Multi-agent Systems». Em: *IEEE Transactions on Power Systems* 22.4, pp. 1753–1759. issn: 0885-8950. doi: 10.1109/TPWRS.2007.908472. url: http://ieeexplore.ieee.org/document/4349107/.

McCarthy, John et al. (2006). «A Proposal for the Dartmouth Summer Research Project on Artificial Intelligence, August 31, 1955». Em: *AI Magazine* 27, pp. 12–14.

Mets, Kevin, Juan Aparicio Ojea e Chris Develder (2014). «Combining Power and Communication Network Simulation for Cost-Effective Smart Grid Analysis». Em: *IEEE Communications Surveys & Tutorials* 16.3, pp. 1771–1796. issn: 1553-877X. doi: 10.1109/SURV.2014.021414.00116. url: http://ieeexplore.ieee.org/document/6766918/.

Minsky, Marvin e Marvin (1986). *The society of mind*. Simon e Schuster, p. 339. isbn: 0671607405. url: https://dl.acm.org/citation.cfm?id=22939.

Nguyen, Van et al. (2017). «On Conceptual Structuration and Coupling Methods of Co-Simulation Frameworks in Cyber-Physical Energy System Validation». Em: *Energies* 10.12, p. 1977. issn: 1996-1073. doi: 10.3390/en10121977. url: http://www.mdpi.com/1996-1073/10/12/1977.

Nicola, Susana, Eduarda Pinto Ferreira e J. J. Pinto Ferreira (2012). «A novel framework for modeling value for the customer, and essay on negotiation». Em: *International Journal of Information Technology & Decision Making* 11.03, pp. 661–703. issn: 0219-6220. doi: 10.1142/S0219622012500162. url: http://www.worldscientific.com/doi/abs/10.1142/S0219622012500162.

Oliveira, Pedro et al. (2012). «MASGriP a multi-agent smart grid simulation platform». Em: *IEEE Power and Energy Society General Meeting*. IEEE, pp. 1–8. isbn: 9781467327275. doi: 10.1109/PESGM.2012.6345649. url: http://ieeexplore.ieee.org/document/6345649/.

OpenDSS (2018). *Simulation Tool – OpenDSSg*. Acedido em fevereiro 2018. url: http://smartgrid.epri.com/SimulationTool.aspx.

Panait, Liviu e Sean Luke (2005). «Cooperative Multi-Agent Learning: The State of the Art». Em: *Autonomous Agents and Multi-Agent Systems* 11.3, pp. 387–434. issn: 1387-2532. doi: 10.1007/s10458-005-2631-2. url: http://link.springer.com/10.1007/s10458-005-2631-2.

Pinto, Tiago et al. (2014). «Adaptive learning in agents behaviour: A framework for electricity markets simulation». Em: *Integrated Computer-Aided Engineering* 21.4, pp. 399–415. issn: 1069-2509. doi: 10.3233/ICA-140477. url: https://dl.acm.org/citation.cfm?id=2691129.

PNNL (2018). *GridLAB-D*. Acedido em fevereiro 2018. url: https://www.gridlabd.org.

Poslad, Stefan, Phil Buckle e Rob Hadingham (2000). «The FIPA-OS Agent Platform: Open Source for Open Standards». Em: *In Proceedings of the 5th International Conference and Exhibition on the Practical Application of Intelligent Agents and Multi-Agents* 355, p. 368. url: https://pdfs.semanticscholar.org/0822/fd0cab13af50d683692c4f76f38059e2be06.

pdf%20http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.146.2570%7B%5C&%7Drep=rep1%7B%5C&%7Dtype=pdf.

Poslad, Stefan e Patricia Charlton (2001). «Standardizing Agent Interoperability: The FIPA Approach». Em: pp. 98–117. url: http://www.eecs.qmul.ac.uk/%7B~%7Dstefan/publications/2001-LNCS-standardising-agent-interoperability.pdf.

PQ Soft (2018). *PSCAD: Transients Simulation Software by the Manitoba HVDC Research Centre*. Acedido em fevereiro 2018. url: http://www.pqsoft.com/pscad/index.htm.

Rousset, Alban et al. (2014). «A Survey on Parallel and Distributed Multi-Agent Systems». Em: *Euro-Par 2014: Parallel Processing Workshops*. Vol. 8805. Springer, pp. 371–382. doi: 10.1007/978-3-319-14325-5_32. url: https://hal.archives-ouvertes.fr/hal-01230768.

Saaty, Thomas L. (1990). «How to make a decision: The analytic hierarchy process». Em: *European Journal of Operational Research* 48.1. Desicion making by the analytic hierarchy process: Theory and applications, pp. 9–26. issn: 0377-2217. doi: https://doi.org/10.1016/0377-2217(90)90057-I. url: http://www.sciencedirect.com/science/article/pii/037722179090057I.

Santos, Gabriel (2015). *Ontologies for the interoperability of multiagent electricity markets simulation platforms*.

Santos, Gabriel, Tiago Pinto, Hugo Morais et al. (2015). «Multi-agent simulation of competitive electricity markets: Autonomous systems cooperation for European market modeling». Em: *Energy Conversion and Management* 99, pp. 387–399. issn: 0196-8904. doi: 10.1016/J.ENCONMAN.2015.04.042. url: https://www.sciencedirect.com/science/article/pii/S0196890415003969.

Santos, Gabriel, Tiago Pinto, Isabel Praça et al. (2016a). «An Interoperable Approach for Energy Systems Simulation: Electricity Market Participation Ontologies». Em: *Energies* 9.11, pp. 1–22. url: https://ideas.repec.org/a/gam/jeners/v9y2016i11p878-d81500.html.

– (2016b). «MASCEM: Optimizing the performance of a multi-agent system». Em: *Energy* 111, pp. 513–524. issn: 0360-5442. doi: https://doi.org/10.1016/j.energy.2016.05.127. url: http://www.sciencedirect.com/science/article/pii/S0360544216307654.

Santos, Gabriel, Tiago Pinto, Zita Vale et al. (2016). «Electricity Markets Ontology to Support MASCEM's Simulations». Em: *Highlights of Practical Applications of Scalable Multi-Agent Systems. The PAAMS Collection*. Ed. por Javier Bajo et al. Cham: Springer International Publishing, pp. 393–404. isbn: 978-3-319-39387-2.

Scherfk, Stefan (2018). *Mosaik Documentation*. Acedido em Fevereiro 2018. url: https://media.readthedocs.org/pdf/mosaik/latest/mosaik.pdf.

Schleiffer, Ralf (2005). «An intelligent agent model». Em: *European Journal of Operational Research*. Vol. 166. 3. North-Holland, pp. 666–693. doi: 10.1016/j.ejor.2004.03.039. url: https://www.sciencedirect.com/science/article/pii/S0377221704004096.

Schütte, Steffen (2013). «Simulation Model Composition for the Large-Scale Analysis of Smart Grid Control Mechanisms». Tese de doutoramento. Oldenburg, p. 407. url: http://oops.uni-oldenburg.de/1768/1/schsim13.pdf.

Schutte, Steffen, Stefan Scherfke e Martin Troschel (2011). «Mosaik: A framework for modular simulation of active components in Smart Grids». Em: *2011 IEEE First International Workshop on Smart Grid Modeling and Simulation (SGMS)*. IEEE, pp. 55–60. isbn: 978-1-4673-0195-4. doi: 10.1109/SGMS.2011.6089027. url: http://ieeexplore.ieee.org/document/6089027/.

Shahidehpour, Mohammad, Hatim Yamin e Zuyi Li (2002). *Market Overview in Electric Power Systems*. John Wiley {&} Sons, Inc., pp. 1–20. isbn: 9780471224129. doi: 10.1002/047122412X.ch1.

Siebert, Julien (2011). «Approche multi-agent pour la multi-modélisation et le couplage de simulations . Application à l'étude des influences entre le fonctionnement des réseaux ambiants et le comportement de leurs utilisateurs». Tese de doutoramento, p. 155. url: https://tel.archives-ouvertes.fr/tel-00642034.

Steinbrink, C et al. (2017). «Simulation-based Validation of Smart Grids – Status Quo and Future Research Trends». Em: arXiv: arXiv:1710.02315v1. url: https://arxiv.org/pdf/1710.02315.pdf.

Stone, Peter e Manuela Veloso (2000). «Multiagent systems: a survey from a machine learning perspective». Em: *Autonomous Robots* 8.3, pp. 345–383. issn: 09295593. doi: 10.1023/A:1008942012299. url: http://link.springer.com/10.1023/A:1008942012299.

Studer, Rudi, V.Richard Benjamins e Dieter Fensel (1998). «Knowledge engineering: Principles and methods». Em: *Data & Knowledge Engineering* 25.1-2, pp. 161–197. issn: 0169-023X. doi: 10.1016/S0169-023X(97)00056-6. url: https://www.sciencedirect.com/science/article/pii/S0169023X97000566.

Tolk, Andreas e James Muguira (2003). «The Levels of Conceptual Interoperability Model». Em: *Fall Simulation Interoperability Workshop* September, pp. 1–9. url: https://pdfs.semanticscholar.org/f655/af160f630b9be8dbab986f6a96953aa3e986.pdf.

Turing, A. M. (1950). *Computing Machinery and Intelligence*. url: http://cogprints.org/499/.

University of Southern California (2018). *The Network Simulator - ns-2*. Acedido em fevereiro 2018. url: https://www.isi.edu/nsnam/ns/.

W3C (2004). *Resource Description Framework*. Acedido em fevereiro 2018. url: https://www.w3.org/RDF/.

Weiss, Gerhard (1999). *A Modern Approach to Distributed Artificial Intelligence*. Ed. por Gerhard Weiss. Cambridge, MA, USA: MIT Press. isbn: 0-262-23203-0.

– (2000). «Distributed Artificial Intelligence Meets Machine Learning: Learning in Multi-Agent Environments.» Em: 3.

Woodall, Tony (2003). «Conceptualising 'Value for the Customer': An Attributional, Structural and Dispositional Analysis». Em:

Wooldridge, Michael e Nicholas R Jennings (1995). «Intelligent Agents: Theory and Practice». Em: *Knowledge Engineering Review* 10, pp. 115–152.

Wüstenhagen, Rolf e Emanuela Menichetti (2012). «Strategic choices for renewable energy investment: Conceptual framework and opportunities for further research». Em: *Energy Policy* 40, pp. 1–10. issn: 0301-4215. doi: http://dx.doi.org/10.1016/j.enpol.2011.06.050. url: http://www.sciencedirect.com/science/article/pii/S0301421511005064.