Infra-estrutura de software baseada em standards de apoio à eficiência energética utilizando fontes de energia renovável

por

Vasco Miguel Delgado Gomes

Dissertação apresentada na Faculdade de Ciências e Tecnologia da Universidade Nova de Lisboa para a obtenção do grau de Mestre em Engenharia Electrotécnica e de Computadores

Júri:

Presidente:

Doutor Luís Filipe dos Santos Gomes (arguente)

Vogais:

Doutor Celson Pantoja Lima (orientador)

Doutor João Francisco Alves Martins (co-orientador)

Doutor João Miguel Murta Pina (arguente)

Fevereiro 2011

Copyright

Infra-estrutura de software baseada em standards de apoio à eficiência energética utilizando fontes de energia renovável

Dedico esta dissertação à minha Família

Agradecimentos

Gostaria de demonstrar a minha gratidão, em primeiro lugar ao meu orientador Doutor Celson Lima e ao meu co-orientador Doutor João Martins, pela ideia desta dissertação, pelo apoio, pela troca de ideias, motivação e pelo tempo disponibilizado, para eles o meu sincero agradecimento e um até já.

Não podia deixar de agradecer ao Pedro Pereira, Gonçalo Cândido e Luís Ribeiro, pela disponibilidade demonstrada e pela troca de ideias, que me auxiliaram na execução desta dissertação.

Aos meus queridos pais por me terem feito o que eu sou hoje, pela paciência, contribuição, tolerância, por todo o suporte emocional e financeiro, sem os quais tudo isto não teria sido possível, um especial obrigado.

À minha esposa, Andreia Silva, pelo carinho, pela paciência de me ouvir mesmo quando não fazia a menor ideia do que eu estava a falar, pela compreensão de não poder estar com ela, o meu especial obrigado e um pedido: um pouco mais de paciência pela nova jornada que irá agora começar.

Por último, mas não menos importante, um muito obrigado a todos os meus amigos que me acompanharam nestes anos de estudo, não vou referir nomes para não me esquecer de ninguém, eles sabem quem são.

Abstract

Energy efficiency is one of the greatest challenges of our days. Presently, all actors involved in the generation, distribution, and consumption of energy are not only concerned but also acting towards a more rational and efficient use of energy. The awareness of the importance of renewable energy sources assume in the current landscape on this matter, becomes critical an efficient management of them.

The current Energy Management Systems (EMSs) have been developed normally targeting new systems, disregarding old facilities or systems that were not designed for this purpose. It thus becomes necessary to develop a tool for most systems and devices, to allow their integration through a set of mechanisms in order to monitor and control.

This thesis aims to develop a Web Services based software infrastructure to integrate new renewable energy sources, in a standards-based software infrastructure, aiming to provide the necessary support to ménage energy related devices (new and legacy), considering an environment where energy is generated, stored, distributed, and consumed in a rational and environmentally correct way.

Such software infrastructure relies on the adoption of two standards, namely Devices Profile for Web Services (DPWS) and the International Electrotechnical Commission's 61850 (IEC 61850) series. The former handles the web services related needs. The latter helps to support the interoperability of Intelligent Electronic Devices (IEDs), through the use of the Abstract Communication Service Interface (ACSI) and the Substation Configuration Language (SCL).

Keywords: Energy Efficiency, Web Services, Service-Oriented Architecture (SOA), DPWS, IEC 61850.

Sumário

A eficiência energética é um dos principais desafios com que nos deparamos. Actualmente, todos os intervenientes na geração, distribuição e consumo de energia, não estão só preocupados, mas também actuando no sentido de um uso mais eficiente e racional da energia. A consciência da importância que as fontes de energia renovável assumem no panorama actual, torna fundamental uma gestão eficiente das mesmas.

Os Energy Management Systems (EMSs) actuais foram desenvolvidos direccionados aos novos sistemas, não tendo em conta instalações antigas ou sistemas que não foram desenhados com essa finalidade. Torna-se assim necessário o desenvolvimento de uma ferramenta para todos os sistemas e dispositivos, de modo a permitir a integração dos mesmos através de um conjunto de mecanismos, com o objectivo de monitorizar e controlar.

Esta dissertação tem como objectivo o desenvolvimento de um software baseado em *Web Services* para integrar novas fontes de energia renovável, numa infra-estrutura de software baseada em *standards*, com o objectivo de prestar o apoio necessário para gerir dispositivos (novos e *legacy*) relacionados com energia, considerando um ambiente onde a energia é gerada, armazenada, distribuída e consumida de uma forma racional e ambientalmente correcta.

Esta infra-estrutura de software conta com a adopção de dois *standards*, designadamente o Devices Profile for Web Services (DPWS) e International Electrotechnical Commission's 61850 (IEC 61850). O primeiro trata dos aspectos relacionados com os serviços orientados, ao passo que o último ajuda a suportar a interoperabilidade de Intelligent Electronic Devices (IEDs), através do uso da Abstract Communication Service Interface (ACSI) e da Substation Configuration Language (SCL).

Palavras-chave: Eficiência Energética, *Web Services*, Service-Oriented Architecture (SOA), DPWS, IEC 61850.

Índice

Lista de Figuras

Lista de Tabelas

Abreviaturas

ACSI	Abstract Communication Service Interface
AVAC	Aquecimento, Ventilação e Ar Condicionado
DA	Data Attribute
DER	Diagrama de Entidade Relacionamento
DIEMS	Distributed Intelligent Energy Management System
DO	Data Object
DPWS	Devices Profile for Web Services
EMS	Energy Management System
EGX	Ethernet Gateway
FCDA	Functional Constrained Data Attribute
HTTP	HyperText Transfer Protocol
ICT	Information and Communication Technology
IEC 61850	International Electrotechnical Commission's 61850
IED	Intelligent Electronic Device
IP	Internet Protocol
LAN	Local Area Network
LD	Logical Device

LN	Logical Node
NEMO	NEtworked MOnitoring & COntrol, Diagnostic for Electrical Distribution
NEMO EN	NEMO Energy Network
NEMO ES	NEMO External Service
NEMO IS	NEMO Internal Service
NEMO SN	NEMO Software Network
NEMO-A	NEMO-Api
NEMO-B	NEMO-Bus
NEMO-C	NEMO-Connector
NEMO-F	NEMO-File
NEMO-K	NEMO-Kernel
NEMO-V	NEMO-Viewer
OSI	Open Systems Interconnection
PD	Physical Device
PLC	Programmable Logic Controller
QREN	Quadro de Referência Estratégico Nacional
RS-232	Recommended Standard 232
RS-485	Recommended Standard 485
RTU	Remote Terminal Unit
SCADA	Supervisory Control And Data Acquisition
SCL	Substation Configuration Language
SOA	Service-Oriented Architecture

SOAP	Simple Object Acess Protocol
TCP/IP	Transmission Control Protocol/Internet Protocol
UDP	User Datagram Protocol
UML	Unified Modeling Language
WSDL	Web Service Definition Language
XML	eXtensible Markup Language

Uso de Termos Oriundos de Outras Línguas

Devido a esta dissertação estar incluída numa área tecnológica onde os termos anglo-saxónicos têm grande predominância e já terem sido publicados artigos com os mesmos, foi decido acrescentar a tabela 1, com os referidos termos. Estes estarão em itálico para que se possam distinguir no documento.

Tabela 1: Termos anglo-saxónicos usados nesta dissertação.

Termo	**Descrição**
Endpoint	Ponto de entrada para um serviço ou um processo.
Ethernet	Tecnologia de interconexão para redes locais.
Fieldbus	Sistema de rede de comunicação industrial para controlo em tempo real.
Legacy	Tecnologia antiga, sistemas de computadores, ou aplicações que continuam a ser usadas, normalmente porque ainda desempenham as funções que o utilizador necessita, apesar de já existirem novas tecnologias e métodos mais eficazes para executar as mesmas funções.
Plug and Play	Permite que o computador reconheça e configure automaticamente qualquer dispositivo que seja instalado, facilitando a expansão segura dos computadores e eliminando a configuração manual.
Standards	O plural de standard.
Web Service	Serviço que permite a integração e normaliza os recursos disponibilizados entre aplicações. A descrição mais pormenorizada pode ser consultada na subsecção 2.2.1.

Capítulo 1

Introdução

Este capítulo tem como objectivo enquadrar o leitor no tema desta dissertação desenvolvida, incluindo introdução, visão global e contexto de desenvolvimento deste trabalho. Por último irá ser apresentado ao leitor a estrutura deste documento.

1.1 Enquadramento do Problema

A era industrial, provocada pela revolução industrial em meados do século XVIII, foi desencadeada por uma série de factores, nomeadamente: grande oferta de matérias-primas (ferro, madeira, petróleo, carvão, etc.); grande desenvolvimento tecnológico (primeira bateria, telefone, lâmpada eléctrica, primeiro veículo com motor a gasolina e também um dos grandes impulsionadores que levou à industrialização, o motor a vapor); a maior disponibilidade de força de trabalho, etc...

O crescimento económico gerado pela industrialização e pela produção em massa, fez com que aparecesse uma sociedade que exigia produtos de baixo custo e de confiança.

Na época acreditava-se que a produção em massa iria impulsionar a humanidade a um desenvolvimento e sofisticação sem precedentes, mas isso não se veio a verificar devido a diversos factores: o avanço tecnológico e a ganância dos lucros aumentou o desemprego, o que veio provocar inúmeros problemas sociais, as despesas de saúde e segurança foram elevadas e o consumidor começou a ficar cada vez mais exigente

em relação a bens personalizados, aumentando assim a importância do bem-estar e satisfação dos consumidores.

Foram tempos de destruição maciça dos recursos naturais e de gestão descuidada da energia, sem nenhuma preocupação ambiental. Os resultados desse consumo imprudente e exagerado de energia são bem conhecidos na actualidade. Embora mais tarde que o esperado, os sentimentos de sensibilização ambiental e de sustentabilidade, são a tendência dos diversos intervenientes da complexa rede energética (incluindo também o consumidor final), para uma utilização mais racional da energia.

A eficiência energética é um dos pilares da sustentabilidade. O aumento de fontes de energia renovável, suportado e reforçado pelas regras do governo, juntamente com os incentivos dos fornecedores de energia, reflectem a seriedade do assunto e ajudam a traçar o panorama actual [20].

1.2 Objectivo da Dissertação

Esta dissertação tem como objectivo principal o desenvolvimento de um sistema computacional baseado nos *standards*: Devices Profile for Web Services (DPWS) e International Electrotechnical Commission's 61850 (IEC 61850), que permitirá a integração de dispositivos relacionados com energia, numa rede de software. Como objectivos específicos pretende-se que o sistema permita descobrir e comunicar com os dispositivos que se encontram conectados na rede de energia e também que permita a monitorização e actuação nos dispositivos.

Nem todos os dispositivos existentes actualmente permitem a sua integração numa rede de software, devido aos fabricantes usarem diferentes protocolos e *standards*. Isto provoca dificuldades quando se quer integrar dispositivos que realizem operações nas diferentes áreas relacionadas com a energia.

Pretende-se então criar uma rede de software, onde os dispositivos se encontram conectados e possam trocar informação, com o objectivo de monitorizar e controlar uma rede de dispositivos relacionados com energia (medidores de energia, relés, etc.), de modo a ser possível comunicar com os dispositivos através de um software baseado em *standards*, denominado NEMO-Connector (NEMO-C).

1.3 Visão da Dissertação

Hoje em dia, o problema da escassez de recursos fósseis preocupa cada vez mais a humanidade, como tal tem-se verificado um aumento significativo da utilização da chamada "energia verde", advindo de uma maior consciencialização das pessoas em relação ao meio ambiente.

Poupar energia não é só consumir menos, mas também consumir de uma forma mais eficiente e racional de modo a desperdiçar a menor quantidade de energia. Quando a energia é produzida, se não for logo gasta ou armazenada, é perdida. Como tal torna-se necessário a eficiente gestão de energia, em diversas áreas: consumo, produção e distribuição.

Será de esperar que a tecnologia se traduza num ferramenta que permita auxiliar a eficiência energética em sistemas, ou conjunto de sistemas onde exista consumo, produção e distribuição de energia.

O conceito desta dissertação assenta no facto de uma rede de dispositivos relacionados com energia, estes poderem ser operados com a ajuda de uma infra-estrutura (distribuída) de software.

Com este conceito torna-se necessário cumprir os objectivos descritos no subcapítulo anterior, deste modo, será possível uma diminuição da energia consumida, porque tudo será efectuado de uma forma mais racional e ambientalmente correcta, devido à monitorização e ao controlo inteligente.

1.4 Contexto de Desenvolvimento

Esta dissertação foi desenvolvida no âmbito do projecto NEtworked MOnitoring & COntrol, Diagnostic for Electrical Distribution (NEMO) financiado pelo Quadro de Referência Estratégico Nacional (QREN), em parceria com as empresas portuguesas Critical Software e ATECNIC e também com a empresa internacional Schneider Electric, esta última com especial importância na disponibilização de componentes eléctricos, através do qual foi possível a instalação da configuração experimental desta dissertação.

O projecto NEMO tem como objectivo o desenvolvimento de uma infra-estrutura de software, suportada pelo paradigma Service-Oriented Architecture (SOA), baseada em *standards*, para a gestão de complexos sistemas distribuídos de energia, onde a produção, distribuição e consumo de energia é considerado. O conceito do projecto NEMO tem como fundamento o facto de uma rede de dispositivos relacionados com energia, estes poderem ser operados com a ajuda de uma infra-estrutura (distribuída) de software.

A contribuição desta dissertação no projecto NEMO, consiste na integração dos dispositivos relacionados com energia na infra-estrutura de software, de modo a ser possível controlar, monitorizar e descobrir os dispositivos que se encontram conectados à rede energia.

1.5 Organização do Documento

Esta dissertação está estruturada em cinco capítulos, nomeadamente:

1. **Introdução:** É apresentado ao leitor o enquadramento, objectivo e visão do projecto, com o intuito de clarificar e motivar o leitor para a leitura do resto do documento.

2. **Estado da Arte:** Este capítulo apresenta uma abordagem geral sobre o estado da arte nas áreas abrangentes da dissertação, nomeadamente, sistemas de gestão de energia e interoperabilidade.

3. **Requisitos e Modelo Conceptual:** Este capítulo apresenta ao leitor os requisitos e o modelo conceptual que suportam a dissertação. Adicionalmente, também fornece um conjunto de definições básicas utilizadas nesta dissertação e descreve o contexto da mesma.

4. **Implementação:** Este capítulo apresenta ao leitor a implementação efectuada na dissertação. Descreve as tecnologias utilizadas, a estrutura e organização do sistema e, por fim, um exemplo de utilização.

5. **Conclusões:** Neste capítulo estão expostas as conclusões retiradas da dissertação e são referidas algumas indicações sobre trabalhos futuros que poderão vir a ser implementados.

Capítulo 2

Sistemas de Gestão de Energia e Interoperabilidade

Este capítulo resume o actual estado da arte das principais áreas envolvidas nesta dissertação. Está dividido em duas secções: Energy Management Systems (EMSs) e Interoperabilidade, que são descritas nas secções 2.1 e 2.2, respectivamente.

2.1 EMSs

Actualmente, o controlo automático de energia tornou-se uma prática normal. Praticamente todos os edifícios não residenciais têm controladores automáticos, com um computador como processador central [40]. Esses sistemas são denominados de Energy Management System (EMS). Actualmente, os proprietários dos edifícios e de fábricas devem regularmente realizar a gestão de energia, avaliar os sistemas informáticos existentes, avaliar as opções dos contratos de serviços e optimizar as operações dos EMSs.

Muitas das funcionalidades avançadas do EMS são subutilizadas. Por exemplo, as capacidades de monitorização e controlo dos EMSs, são ferramentas poderosas para melhorar o consumo do Aquecimento, Ventilação e Ar Condicionado (AVAC) e da

iluminação e na redução da utilização de energia, mas a maioria dos gestores de instalações e os operadores dos sistemas, simplesmente não tem tempo para investigar estes recursos.

O problema da gestão energética e dos custos da gestão de energia sempre foi um problema para muitos utilizadores comerciais que operam grandes instalações físicas (isto é, instalações e/ou fábricas), por causa da grande quantidade de energia consumida pelas mesmas. É aconselhável gerir e analisar o consumo de energia das instalações, a fim de reduzir os custos totais de energias das instalações. O processo de gestão de energia pode envolver várias etapas, como por exemplo, comprar energia de um fornecedor mais barato, ou ajustar o consumo de energia da instalação para períodos onde a energia é mais barata.

A subsecção 2.1.1 faz uma introdução ao sistema Supervisory Control And Data Acquisition (SCADA). De seguida, na subsecção 2.1.2 apresenta-se os EMSs disponíveis actualmente. Por fim, irá ser apresentada a subsecção 2.1.3, com os diferentes domínios onde os EMSs tem sido aplicados, bem como os resultados alcançados.

2.1.1 SCADA

Os sistemas SCADA são largamente usados nos processos industriais de supervisão, controlo e aquisição de dados [12]. As empresas que fazem parte do comité de normalização estão, portanto, a estabelecer as tendências das Information and Communication Technologies (ICTs), que geralmente desenvolvem esses sistemas. Os sistemas SCADA realizaram progressos significativos nos últimos anos, em termos de funcionalidade, desempenho, escalabilidade e abertura, de tal forma que eles são uma alternativa para desenvolvimento habitacional, mas também para sistemas de controlo complexo, tais como experiências físicas.

2.1.1.1 Definição de SCADA

Os sistemas SCADA são o suporte para a supervisão, controlo e aquisição de dados, sendo largamente utilizados nos processos industriais de supervisão, controlo e aquisição de dados, contudo não são sistemas de controlo completos, centrando-se ao nível de

supervisão. O SCADA é utilizado na maioria dos processo industriais (por exemplo, siderurgia, geração e distribuição de energia, química), sendo também actualmente utilizado devido à sua fiabilidade em algumas instalações experimentais, tais como na fusão nuclear. Um sistema SCADA pode variar desde um computador a recolher dados, ou em sistemas maiores a estação principal pode ser composta por vários servidores, aplicações de software distribuídas e sites de recuperação de calamidades [36].

2.1.1.2 Gerações dos sistemas SCADA

Os primeiros sistemas de "SCADA" realizavam a aquisição de dados através de contadores, luzes e gráficos de agulhas que escreviam os valores medidos, em rolos de papel [2]. O operador controlava manualmente o sistema, realizando assim supervisão. Estes sistemas independentes e com apenas um computador pertenciam à primeira geração de sistemas SCADA, denominada Monolítica.

Na segunda geração dos sistemas SCADA, denominada Distribuída, as informações entre as várias estações era partilhada em tempo real através de Local Area Networks (LANs) e o processamento de dados era distribuído entre várias estações, reduzindo o custo e o tamanho das estações devido a cada tarefa ser atribuída a uma estação

A actual geração de sistemas SCADA é a terceira, denominada Em Rede, onde o sistema SCADA em vez de utilizar um ambiente proprietário, usa um sistema de arquitectura aberto, permitindo assim a conectividade de qualquer dispositivo periférico ao sistema. As comunicações com a estação principal é realizada através de Internet Protocol (IP)

2.1.1.3 Arquitectura de um sistema SCADA

Um sistema SCADA é composto por vários Remote Terminal Units (RTUs) que realizam a recolha de dados e os enviam para uma estação principal, através de um sistema de comunicação. A estação principal exibe os dados adquiridos e permite ao operador efectuar tarefas através de controlo remoto.

Os dados recolhidos permitem à estação principal a optimização de operações, tornando-as mais eficientes, mais confiáveis e também mais segura, resultando numa operação

com menor custo, em relação a sistemas não automatizados. Um sistema SCADA é composto por:

- Aparelho que apresenta ao operador os dados do sistema e que também permite monitorizar e controlar.
- Um sistema de supervisão que adquire todos os dados necessários sobre o processo.
- RTUs que estão conectados aos sensores, recolhendo informação que enviam para o sistema de supervisão.
- Programmable Logic Controllers (PLCs) que são usados como dispositivos de recolha de dados em vez dos RTUs porque são mais versáteis, mais facilmente configuráveis e mais económicas.
- A infra-estrutura de comunicação que conecta os RTUs ao sistema de supervisão.

O RTU fornece uma interface para a área analógica e para os sensores digitais, situados em cada local remoto, convertendo todos os sinais eléctricos do equipamento em valores digitais, como o estado: aberto/fechado; medições: fluxo, pressão, corrente ou tensão. As estações supervisoras recolhem dados de vários RTUs e, normalmente, fornecem uma interface ao operador, para a visualização de informações e controlo dos locais remotos.

2.1.1.4 Evolução dos sistemas SCADA

Como referido na subsecção 2.1.1.2, os sistemas SCADA sofreram diversas evoluções ao longo dos anos, desde um sistema de monitorização onde um computador está conectado a diversos Intelligent Electronic Devices (IEDs) (figura 2.1), até um sistema mais completo ilustrado na figura 2.2.

Na figura 2.1, um computador está conectado directamente aos IEDs que realizam a monitorização e o controlo dos dispositivo. A conexão é realizada através de um *fieldbus*, tais como Profibus, DeviceNet ou Foundation Fieldbus. Os IEDs incluem a inteligência suficiente para a aquisição de dados, comunicação com outros dispositivos e a realização da sua parte no processo. Cada um destes sensores inteligente pode ter

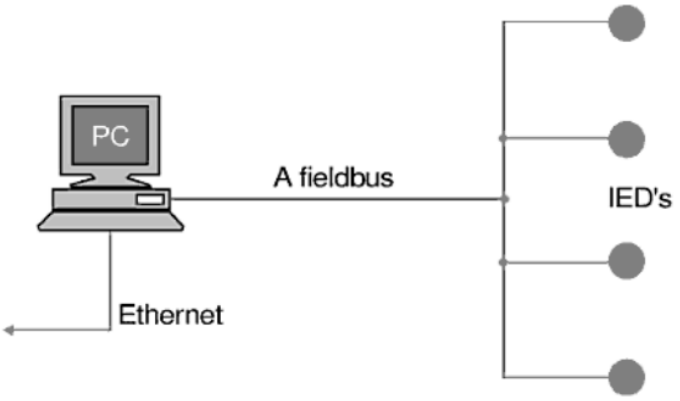

FIGURA 2.1: PC para IED através de *fieldbus* [2].

mais que um sensor embutido. Normalmente, um IED pode combinar um sensor com entrada analógica, uma saída analógica, um controlador, um sistema de comunicação e um programa em memória, tudo isto num único dispositivo. Esta arquitectura tem as seguintes vantagens:

- A quantidade de fios necessária é mínima.
- O operador pode consultar até ao nível do sensor.
- Os dados recebidos do dispositivo podem incluir informação, tais como números de série, modelo, quando ele foi instalado e por quem.
- Todos os dispositivos são *Plug and Play* para serem facilmente instalados e substituídos.
- Dispositivos mais pequenos significam menor espaço para o sistema de aquisição de dados.

Existem algumas desvantagens nesta arquitectura, nomeadamente: torna-se necessário funcionários com melhor formação, o preço dos sensores é mais elevado e a dependência pelo sistema de comunicação é superior.

Como referido anteriormente, a figura 2.2 ilustra um sistema SCADA típico, sendo este normalmente composto pelas seguintes funcionalidades:

- Interface gráfica.

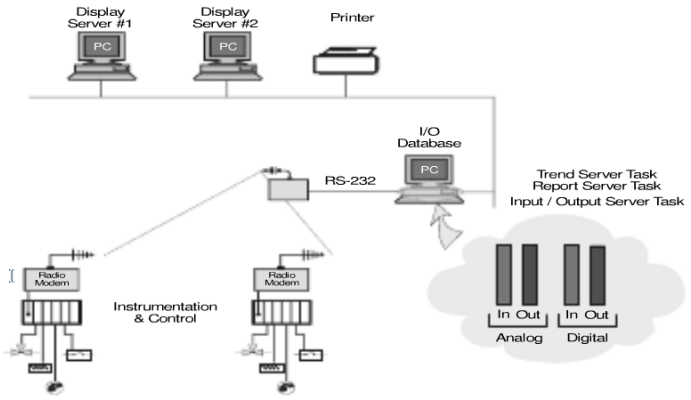

FIGURA 2.2: Sistema SCADA típico actual [2].

- Disponibilização de gráficos.
- Alarmes.
- Interfaces RTU (e PLC).
- Escalabilidade.
- Acesso à informação.
- Base de dados.
- Funcionamento em rede.
- Tolerância a falhas e redundância.
- Processamento distribuído cliente/servidor.

Num sistema SCADA com esta arquitectura, a estação principal pode ser composta por vários servidores, aplicações de software distribuídas e sites de recuperação de calamidades. Para aumentar a integridade do sistema, múltiplos servidores são configurados especificamente para fornecer monitorização e controlo contínuos durante a falha do servidor.

Em alguns sistemas SCADA o hardware é robusto para resistir a tensões, temperaturas e vibrações extremas, mas a fiabilidade é ainda maior em instalações críticas, através da inclusão de canais de comunicação e hardware redundante, até existirem múltiplos centros de controlo que estejam completamente equipados. O equipamento que estiver

defeituoso, pode ser identificado e as funcionalidades podem ser assumidas através de hardware de segurança, que pode ser substituído sem qualquer interrupção do processo.

2.1.2 Os EMSs disponíveis actualmente

Existem três principais razões para uma organização adquirir um EMS (figura 2.3), a primeira razão é reduzir e controlar os custos, de seguida, proteger o clima global e a terceira razão é aumentar a reputação da empresa juntos aos clientes [14].

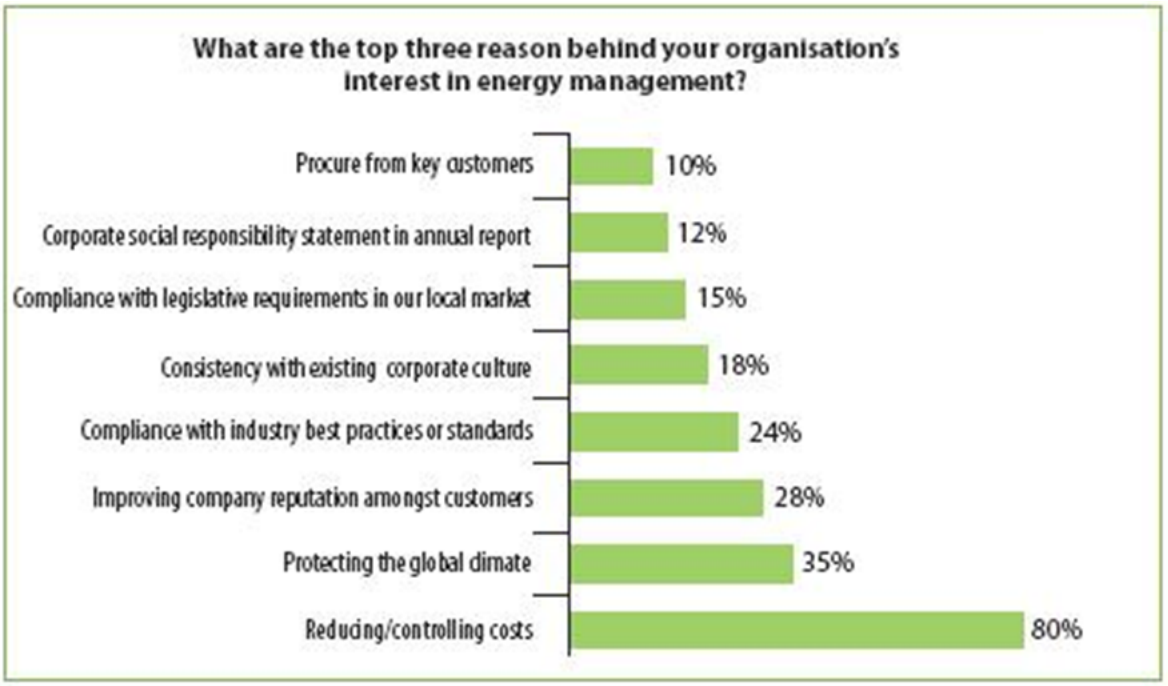

FIGURA 2.3: Resultado de um inquérito realizado às organizações, onde se perguntava quais as três principais razões do interesse num EMS [14].

Uma breve consulta às fontes de informação existentes aponta para várias soluções de EMSs disponíveis no mercado por várias empresas da área. Os EMSs disponíveis podem ir de simples softwares de controlo de facturas até à gestão completa de edifícios ou instalações. Nas subsecções seguintes irá ser exposto ao leitor vários EMSs e algumas das características mais relevantes.

2.1.2.1 *EnergyCap*

O *EnergyCap* permite acompanhar qualquer produto e visualizar detalhadamente por níveis, qualquer tipo de factura [15]. Inclui navegação através de exibição em árvore, memorandos e sistema de mensagens e pode também controlar facturas, permitindo encontrar facturas que não correspondem ao valor que deviam.

O *EnergyCap* torna mais fácil introduzir e recuperar informações sobre qualquer produto (electricidade, gás natural, propano, óleo, água, esgotos, resíduos, reciclagem, telecomunicações, etc.) e qualquer quantidade de contas, medições, facturas e os detalhes das facturas. Ver o custo de cada conta e o histórico do consumo através de gráficos ou tabelas, comparar facilmente o ano fiscal, mensal, ou resumos anuais, e calcular as emissões dos gases de efeito de estufa. Com este software pode-se também beneficiar do processamento rápido de uma factura e pode-se navegar facilmente na aplicação com a funcionalidade da pesquisa automática. Um exemplo da interface da aplicação pode ser visto na figura 2.4.

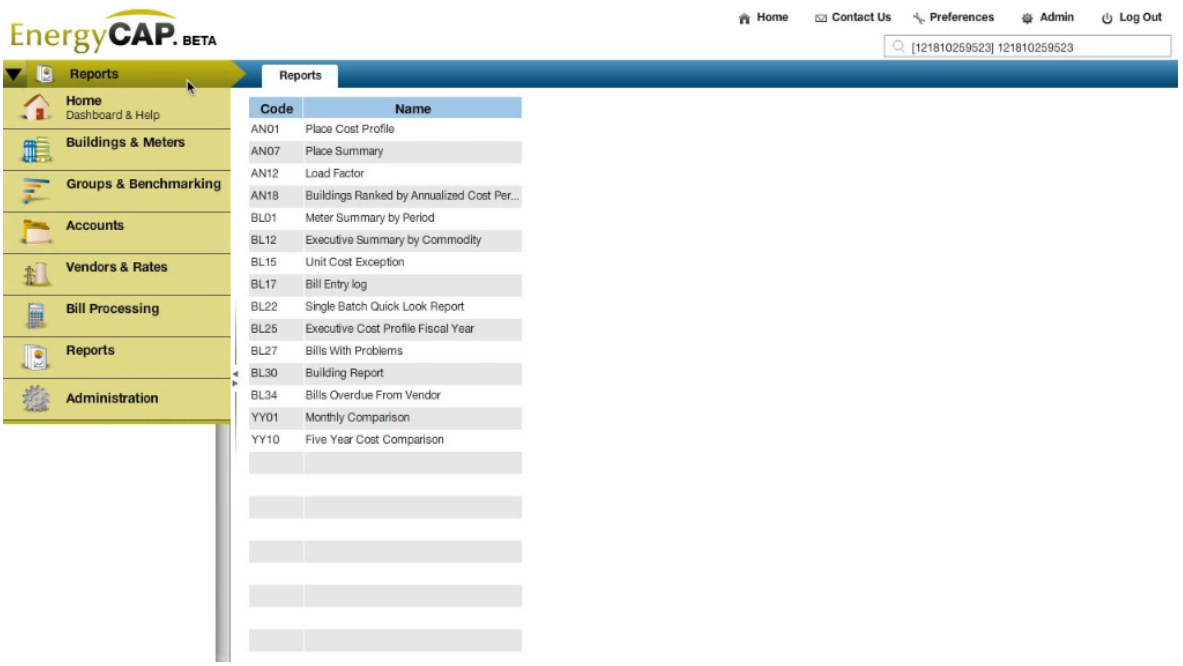

FIGURA 2.4: Interface da aplicação *EnergyCap*.

2.1.2.2 *eComponents*

O EMS *eComponents* indica que o primeiro passo para reduzir os custos e conservar energia começa com ferramentas de monitorização e medida de confiança [16]. Para isso, este EMS disponibiliza as seguintes funcionalidades:

- Medir, controlar e analisar o desempenho energético.
- Estabelecer as referências dos padrões de desempenho.
- Monitorar e realizar um relatório das emissões dos gases de estufa.
- Atribuir os custos de energia a unidades operacionais e inquilinos.
- Previsão da necessidade do fornecimento de energia.
- Apoiar programas de compra de energia.
- Verificar a poupança de programas de eficiência energética.
- Analisar e verificar contas de serviços públicos.

Na figura 2.5 pode-se observar como esta aplicação realiza a monitorização de vários sistemas. O utilizador pode monitorizar quais os seus consumos instantâneos e visualizar quando está a gastar mais que o pretendido.

FIGURA 2.5: Exemplo da monitorização de vários sistemas com o *eComponents* [16].

A figura 2.6 é um exemplo dos vários gráficos que esta aplicação pode gerar. Este gráfico em particular permite visualizar qual o consumo de energia, durante as várias horas do dia e deste modo, saber qual a hora do dia em que se consume mais energia.

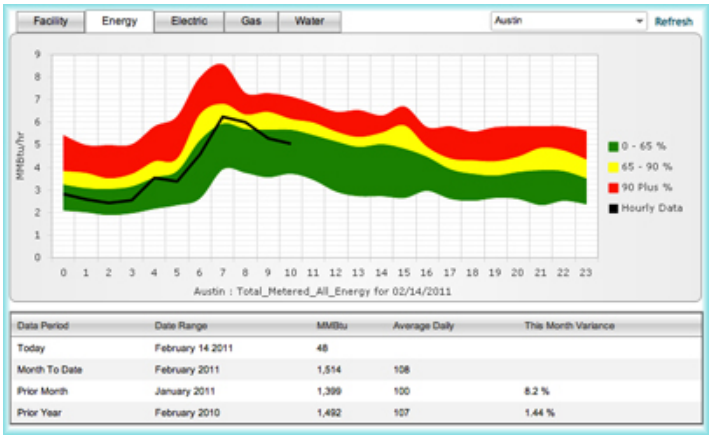

FIGURA 2.6: Exemplo da visualização de gráficos com o *eComponents* [16].

2.1.2.3 *WebLink*

A TEMSCO fornece EMSs que incentivam a poupança para os utilizadores de energia através do controlo de temperatura, AVAC, luz, e o controlo de qualquer tipo de equipamento que necessite de electricidade ou de gás natural [43]. Cada EMS é projectado de acordo com as necessidades dos requisitos de funcionamento e eficiência do cliente.

Este EMS oferece a possibilidade de monitorizar o consumo de energia de uma instalação, com a adição de um elemento de medida com diversos graus de facturação. Tem também a funcionalidade de receber um alerta por e-mail, caso exista um uso anormal de energia. Por último, permite reduzir automaticamente a exigência energética de um edifício, definindo um limite para o efeito.

Na figura 2.7 pode-se observar uma funcionalidade interessante deste EMS, a qual consiste em visualizar o consumo de certas áreas numa instalação.

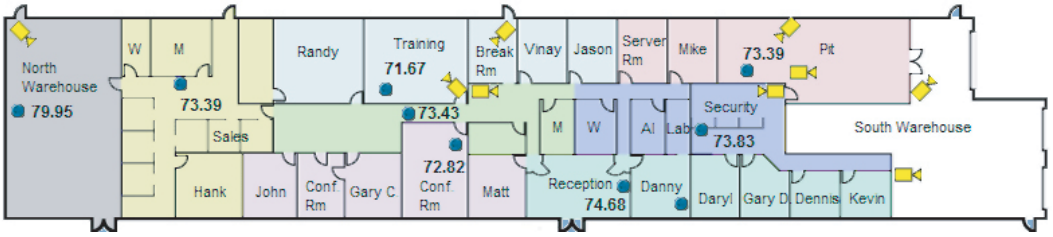

FIGURA 2.7: Exemplo da visualização do consumo com o *WebLink* [43].

2.1.2.4 EMSs da Schneider Electric

A Schneider Electric disponibiliza vários EMSs mais orientados para a gestão de edifícios. Os preços da energia estão a subir, os orçamentos a descer e as escolhas tecnológicas a aumentar. A legislação dos denominados edifícios ecológicos é mais prevalente [41]. Nos Estados Unidos, todos os edifícios públicos têm de ter certificados energéticos que mostrem o que está a ser realizado para reduzir eficientemente o consumo de energia.

Os edifícios precisam de um cérebro inteligente de controlo de todos os sistemas e dos milhares de dados que estes podem gerar. A Schneider Electric disponibiliza esse cérebro, fornecendo uma abordagem integrada que une vários sistemas numa rede entre empresas. Uma vez integrados, os dados desses sistemas são assimilados e transformados em informação que permitem tomar decisões e acções para melhorar o conforto, eficiência e bem-estar para os ocupantes e proprietários do edifício. Este EMS permite a gestão de edifícios inteligentes e sistemas de segurança, que oferecem soluções para climatização, controlo de acesso, vídeo vigilância, controlo de iluminação e eficiência energética.

Um dos softwares mais conhecidos da Schneider Electric é o *Tac Vista*, o qual permite conectar vários dispositivos, independentemente do seu protocolo de comunicação, e interagir com os mesmos. A figura 2.8 mostra um sistema *Tac Vista*, onde se pode observar vários dispositivos conectados numa rede, à qual estão depois conectados vários dispositivos de controlo e gestão de energia.

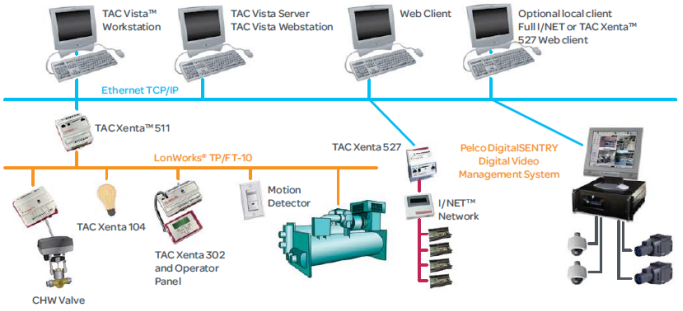

FIGURA 2.8: Arquitectura de um sistema *Tac Vista* [41].

2.1.3 Os diferentes domínios de um EMS

Nos últimos anos, os EMSs têm sido usados em muitos domínios diferentes e sempre com o mesmo objectivo: reduzir o consumo de energia quer seja em automóveis, casas ou fábricas.

Nas subsecções seguintes, irão ser apresentados vários exemplos de utilização dos EMSs.

2.1.3.1 EMS para um veículo eléctrico

Um sistema muito eficiente de gestão de energia para veículos híbridos eléctricos utilizando redes neuronais foi desenvolvido e testado num veículo utilitário [31]. O sistema minimiza a necessidade de energia do veículo e pode funcionar com diferentes fontes de energia primária, como células de combustível, micro turbinas, pilhas de zinco-ar, ou outras fontes de alimentação com pouca habilidade de recuperar energia a partir de uma travagem regenerativa, ou com uma escassa capacidade de poder para uma aceleração rápida. O veículo híbrido eléctrico experimental usa baterias de chumbo-ácido, um ultra-condensador e um motor de corrente contínua sem escovas, com uma potência nominal de 32 kW e uma potência máxima de 53 kW. Na figura 2.9 podemos visualizar a alimentação de um circuito de um típico veículo híbrido.

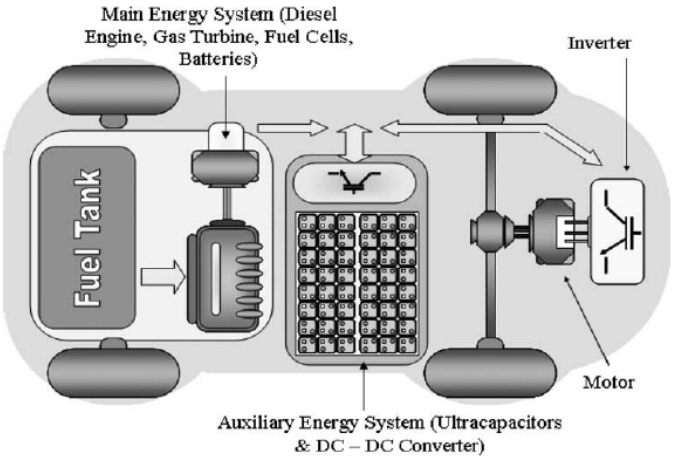

FIGURA 2.9: Alimentação de um circuito de um típico veículo híbrido [31].

O sistema de controlo distribuído mede e armazena os seguintes parâmetros: tensão da fonte primária, a velocidade do carro, corrente instantânea na fonte primária e

no condensador e a actual tensão no condensador. Quando os condensadores foram instalados no veículo, o aumento foi cerca de 5,3% em testes em cidade. No entanto, quando o controlo óptimo das redes neuronais foi usado, este valor aumentou para 8,9%, como se pode verificar na tabela 2.1.

TABELA 2.1: Resultados dos testes de estrada do EMS para um veículo híbrido [31].

	Drive City Circuit (Km)	Kwh Used	Ah Used	Km/KWh	Km/KWh Improvement
Batteries without Regeneration	14.2	5.45	13.90	2.61	-
Batteries with Regeneration	14.2	4.6125	11.23	3.09	18.2%
Batteries with AES (SOC Control)	14.2	4.36	10.55	3.25	24.4%
Batteries with AES (Optimal Neural Network Control)	14.2	4.24	10.58	3.36	28.7%

2.1.3.2 A arquitectura de um EMS proactivo para habitações ecológicas

Os sistemas para casas ecológicas estão a ser desenvolvidos tendo em conta três conceitos importantes como: a casa passiva e activa e as tecnologias de integração [10]. Normalmente os EMSs para habitações são uma técnica activa e de integração da fusão entre tecnologias de energia e ICT. Mas actualmente, a maioria dos EMSs para habitações tem apenas um dispositivo de medição inteligente a interagir com o fornecedor de energia eléctrica. O EMS para habitações proposto (figura 2.10), oferece sistemas de baixo consumo e sem fios, de transferência de dados e de monitorização de energia e vários modelos de utilização de energia com funções de análise e estatística. Além disso, o EMS para habitações monitora a energia eléctrica e a sua utilização, para cada objectivo proposto os dispositivos para habitações ecológicas podem tornar a poupança mais proactiva e ajudar a tornar uma casa mais ecológica.

Conforme mostrado na figura 2.10, os sensores detectam a potência e a energia dos aparelhos eléctricos domésticos e enviam esses dados, através de uma rede sem fios de baixa potência 802.15.4, para o nó coordenador do servidor da casa ecológica. Quando

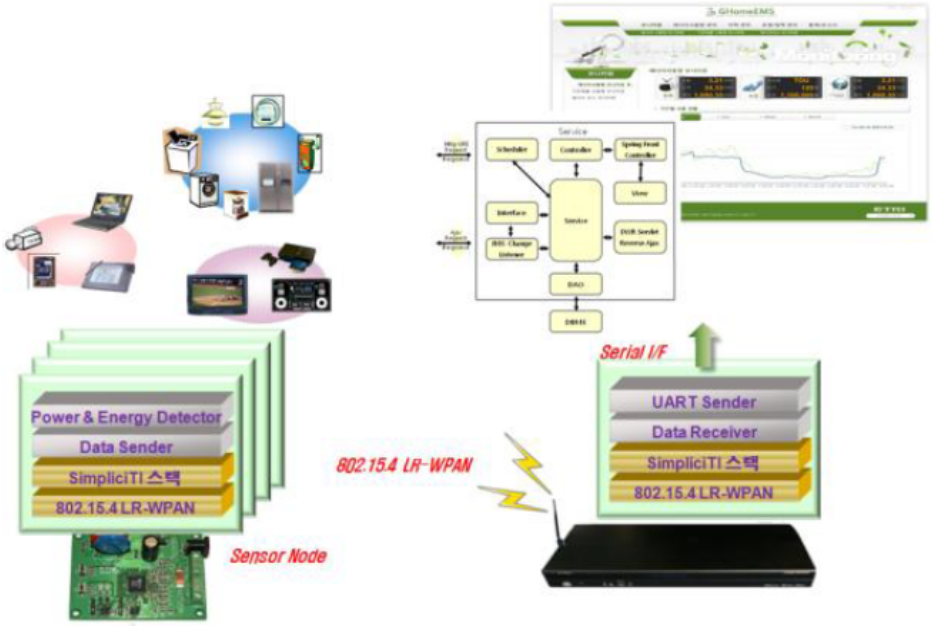

FIGURA 2.10: Arquitectura de um EMS para habitações ecológicas [10].

cada um dos dados de potência e energia são recebidos, são guardados em tempo real no módulo da base de dados.

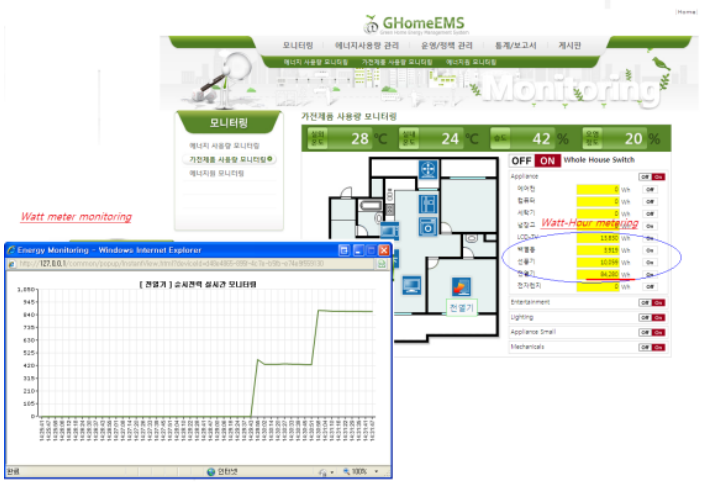

FIGURA 2.11: Modelo estatístico do uso de energia [10].

Na figura 2.11 o utilizador pode seleccionar o dispositivo que quer monitorizar, enquanto na figura 2.12 pode-se observar as funções de análise estatística que mostra a utilização de energia por tipo de aparelho e o tempo de utilização do mesmo. Esta análise ajuda a saber que tipo de aparelho desperdiça mais energia no seu estado inactivo e quanta energia é utilizada no horário de pico. O EMS para habitações ecológicas ajuda o

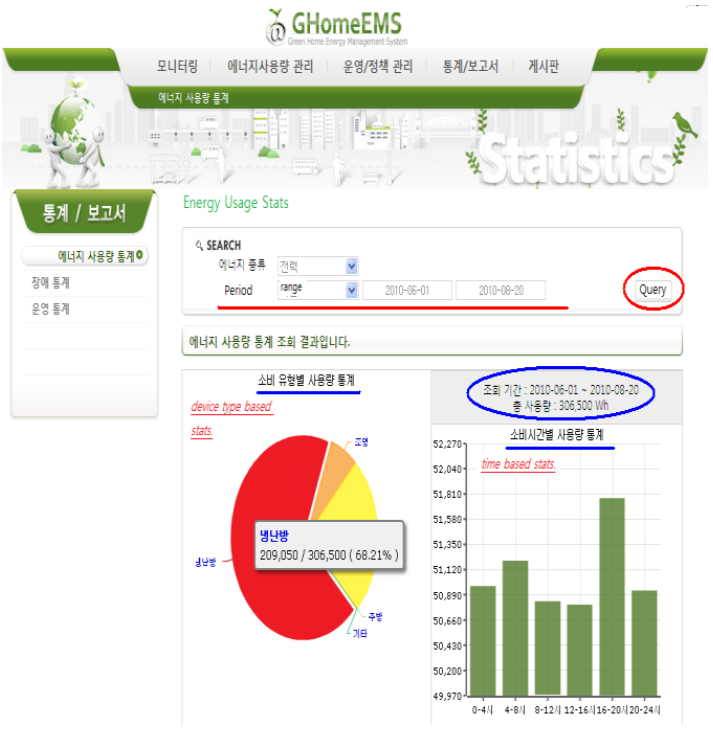

FIGURA 2.12: Energia instantânea e monitorização de energia por dispositivo [10].

utilizador a criar perfis de utilização inteligente de energia e a economizar energia de forma drástica.

2.1.3.3 EMS para redução do consumo do ar condicionado num veículo automóvel

A crescente procura na redução do consumo de energia em veículos está a resultar no desenvolvimento de EMS inteligentes para os veículos, mas também para os seus componentes internos [26].

O sistema de ar condicionado de um veículo é o componente interno que mais contribui para o aumento do consumo total de energia, mas também contribui para o aumento do consumo de combustível.

Para reduzir o consumo do ar condicionado, o EMS vai: controlar o fluxo da massa de ar através do ajuste da velocidade do ventilador; ajudar automaticamente a admissão de ar fresco, bem como a circulação de ar, de modo a maximizar a qualidade de ar no

interior do veículo e, por fim, o controlar do extractor de ar através de um algoritmo inteligente.

Para testar este EMS foram executadas duas simulações nas mesmas condições iniciais. Na simulação 1, o veículo foi testado num dia ensolarado, circulando com o carro a 20 m/s durante 120 segundos. Posteriormente, foi ligado o ventilador assim como o ar condicionado. A simulação 2 foi realizada nas mesmas condições da primeira, mas foi utilizado o EMS.

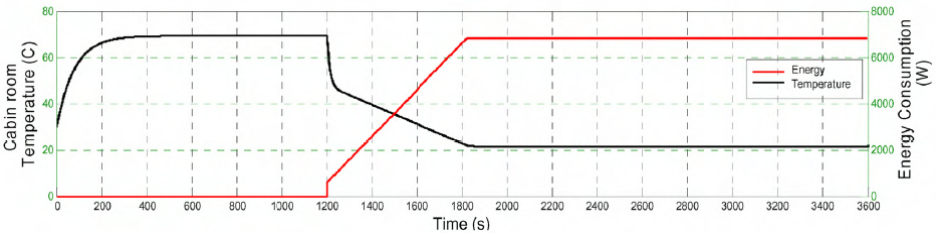

FIGURA 2.13: Simulação 1 : Resultados sem o uso do EMS [26].

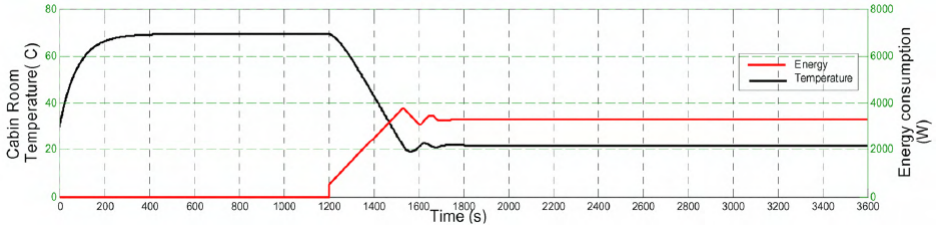

FIGURA 2.14: Simulação 2 : Resultados com o uso do EMS [26].

Através da análise da figura 2.13 e da 2.14, verifica-se que a poupança de energia é notória, mas também se verifica que com um menor gasto de energia consegue-se atingir uma temperatura de conforto mais baixa no interior do veículo.

2.1.3.4 EMS inteligente distribuído para uma micro rede monofásica

Neste caso foi considerada uma micro rede de corrente alternada monofásica de alta frequência, como uma nova solução para integrar num sistema distribuído de geração com fontes de energia renovável [9]. Um sistema Distributed Intelligent Energy Management System (DIEMS) foi implementado para optimizar os custos de operação. Como a optimização depende da energia produzida e sendo esta de fontes de energia renovável, depende fortemente do tempo. Uma rede neuronal foi utilizada para a previsão

meteorológica, para determinar o tipo do dia seguinte. De acordo com o dia previsto, um esquema de optimização é desenvolvido utilizando programação linear juntamente com heurísticas.

Para a simulação, dois sistemas são comparados, um sem DIEMS e outro com DIEMS. Foi considerado um período de 5 dias com várias mudanças diárias consecutivas, de sol para nublado, depois para muito nublado, onde fica durante dois dias e finalmente fica sol no quinto dia. Os resultados da simulação são visíveis na tabela 2.2.

TABELA 2.2: Custos de operação da micro rede com e sem DIEMS [9].

	Day Type	Utility ($)	System ($)	Storage Reserve (kWh)	Number of Hours (<25% Charge)	Savings ($)	Savings (%)	Min. Storage (kWh)	DOD (%)
Without DIEMS	Sunny	$1.45	$0.84	0.10	6	$0.61	42.07%	0.10	95
	Cloudy	$1.23	$1.01	0.61	14	$0.22	17.89%	0.10	95
	Very Cloudy	$1.62	$1.50	0.36	1	$0.12	7.41%	0.36	82
	Very Cloudy	$1.62	$1.55	0.71	13	$0.07	4.32%	0.36	82
	Sunny	$1.45	$0.85	0.56	0	$0.60	41.38%	0.56	72
	Total	$7.37	$5.75		34	$1.62	21.98%		
With DIEMS	Sunny	$1.45	$0.82	1.50	0	$0.63	43.45%	0.50	75
	Cloudy	$1.23	$0.92	1.20	0	$0.31	25.20%	0.90	55
	Very Cloudy	$1.62	$1.52	1.20	0	$0.10	6.17%	0.89	55
	Very Cloudy	$1.62	$1.49	0.63	0	$0.13	8.02%	0.53	73
	Sunny	$1.45	$0.79	0.51	0	$0.66	45.52%	0.51	74
	Total	$7.37	$5.54		0	$1.83	24.83%		

Na tabela 2.2, a coluna *Utility* mostra o custo de energia para um perfil de carga especial, se a micro rede não existir. A coluna *System* dá o custo da energia com a existência da micro rede. *Storage Reserve* indica o estado de carga do armazenamento ao final do dia. *Min. Storag* é o menor estado de carga atingido pela bateria durante a descarga. A coluna *DOD* indica o máximo estado de descarga para armazenamento. É evidente que durante o primeiro dia de sol, o aumento na poupança é pequeno, apenas de 1,38% com o DIEMS em operação. Mas no próximo dia nublado, aumenta a economia em 7,31%. Isto acontece devido ao facto de, com base nas previsões do tipo de dia, o DIEMS sabe que o próximo dia será nublado, escolhendo assim um esquema apropriado de carga e descarga. O aumento da poupança para o intervalo de 5 dias foi cerca de 3 %, o que pode ser considerado significativo, considerando o pequeno sistema aqui usado.

2.2 Interoperabilidade

A interoperabilidade é a capacidade de um sistema, ou produto, de trabalhar com outros sistemas, ou produtos, com o mínimo de esforço possível [21]. A interoperabilidade torna-se uma qualidade de importância crescente para as ICTs, como tal, o termo é amplamente usado na descrição de um produto.

A inovação possibilitada pela interoperabilidade origina amplos benefícios para as sociedades que a fomentam [18]. Os consumidores têm como benefícios: a interoperabilidade conduz à inovação, que resulta em sistemas de tecnologia que trabalham juntos mais facilmente, com menos problemas e assegura que tem mais opções no momento da compra, ou do uso. A inovação que advêm deste tipo de interoperabilidade é vantajosa na medida em que aumenta o número de postos de trabalho e também os índices de produtividade das economias mundiais. A interoperabilidade vai também aumentar a inovação, porque as empresas concorrentes vão desenvolver novos produtos em cima de outros produtos inovadores que foram desenvolvidos anteriormente [45]. Nas secções seguintes, irão ser dados alguns exemplos de interoperabilidade.

2.2.1 *Web Services*

Por definição [6], *Web Service* é um sistema de software desenhado para suportar a interoperabilidade na interacção entre máquinas numa rede. Tem descrito uma interface num formato que é processável por máquinas (especificamente Web Service Definition Language (WSDL)). Outros sistemas interagem com os *Web Services* de um modo definido pela sua descrição, usando mensagens Simple Object Acess Protocol (SOAP), normalmente transmitidas através de HyperText Transfer Protocol (HTTP) com uma serialização eXtensible Markup Language (XML), conjuntamente com outras normas relacionadas com Web.

Ao contrário do modelo tradicional cliente/servidor, tal como o sistema *Web Services*/Página Web, os *Web Services* não fornecem uma interface gráfica ao utilizador, mas permitem a partilha de dados e de processos através de uma rede. Os *Web Services* permitem que aplicações diferentes comuniquem entre si, sem a demorada codificação.

Como toda a comunicação é em XML, os *Web Services* não estão vinculados a nenhum sistema operativo específico ou a uma linguagem de programação. Por exemplo, uma aplicação em Java pode comunicar com uma aplicação em Perl, tal como uma aplicação em Windows pode comunicar com aplicações UNIX.

Existem diversas formas de uma entidade requerente contratar e utilizar um *Web Service*. Em geral, as principais etapas necessárias (figura 2.15) são :

1. As entidades requerentes de serviços e as prestadoras de serviços tornam-se conhecidas uma da outra (ou pelo menos uma conhece o outra).

2. As entidades requerentes e as prestadoras concordam, de alguma forma, com a descrição do serviço e com a semântica que irá reger a interacção entre o requerente e o prestador.

3. A descrição e a semântica do serviço são realizadas pelos agentes requerentes e prestadores.

4. Os agentes requerentes e prestadores trocam mensagens, realizando assim uma tarefa em nome das entidades requerentes e prestadoras. Ou seja, a troca de mensagens com o agente prestador representa a manifestação concreta da interacção com o *Web Service* da entidade prestadora.

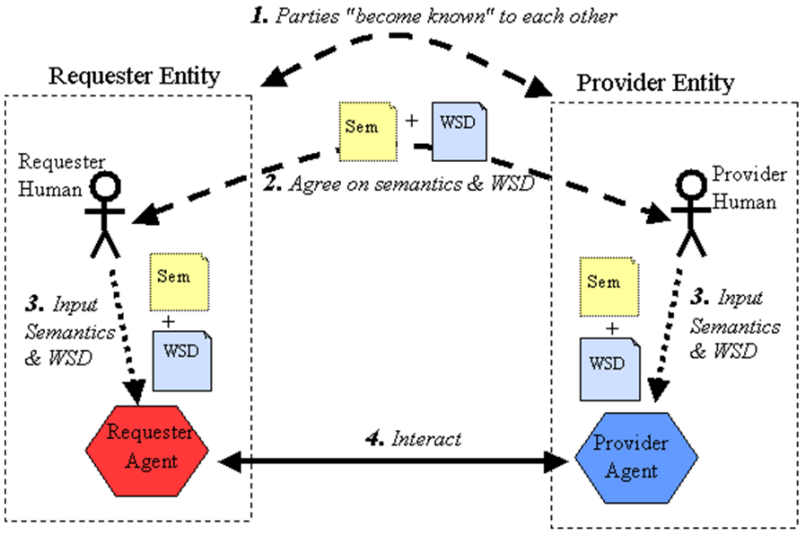

FIGURA 2.15: Processo geral de contratação de um *Web Service* [6].

2.2.2 SOA

SOA é essencialmente uma colecção de serviços, os quais podem comunicar entre si. Essa comunicação pode envolver uma simples transferência de dados ou uma coordenação de dois ou mais serviços [3].

Existem poucas dúvidas que o paradigma SOA já é um tópico importante em muitos ramos da tecnologia [4, 5, 34], não exclusivamente à camada de negócios ICT. A sua influência e adopção está-se a propagar aos mais diferentes domínios de aplicações, tais como, aplicações empresariais, telecomunicações, saúde, transporte, construção e automação industrial.

Por definição [17], SOA estabelece uma arquitectura que visa a aumentar a eficiência, agilidade e produtividade de uma empresa através de serviços básicos que suportam a realização de objectivos estratégicos associados com computação orientada aos serviços.

SOA é um assunto vasto, complexo e multidisciplinar, cuja definição está longe de um consenso (uma procura rápida na literatura facilmente confirma este facto). No entanto, nas várias definições de SOA, encontram-se os seguintes tópicos:

- **Autonomia:** Não há nenhuma dependência directa entre os serviços e eles são estruturalmente dissociados.
- **Interoperabilidade:** É conseguida especificando uma interface que descreve os serviços existentes e o modelo de interacção considerado, em vez de detalhar os serviços desempenhados/prestados pelo prestador de serviços.
- **Independência de plataforma:** Os serviços são idealmente definidos usando formatos baseados em texto (XML [7], WSDL [11], ebXML, etc. . .). Deste modo, a representação dos serviços não está ligada a nenhuma arquitectura de computador, sistema operativo, linguagem de programação ou tecnologia, o que permite que os serviços sejam descodificados por qualquer sistema.
- **Encapsulamento:** Os serviços fornecem funcionalidades independentes que são expostas por interfaces definidas pelo utilizador, escondendo detalhes desnecessários. Serviços complexos são oferecidos através de uma interface simples e limpa, devido à orquestração e à composição de serviços.

- **Disponibilidade/Descoberta:** Os serviços podem ser publicados em registos públicos e disponibilizados para uso geral.

Autonomia e interoperabilidade podem parecer propriedades contraditórias, no entanto, SOA consegue combiná-las sem problemas, uma vez que um serviço é considerado autónomo uma vez que é criado e opera independentemente do seu meio e é auto-suficiente. Também é interoperável através da sua interface que expõem todas as suas funcionalidades para o meio, abstraindo os detalhes de implementação.

Sendo o SOA um paradigma emergente para a modelação de sistemas distribuídos, é muitas vezes confundida com uma vasta gama de ICT em rede. Neste contexto, os *Web Services* são o mecanismo preferido para a implementação de SOA [33]. Em SOA, os aspectos lógicos e físicos de recurso estão desacoplados, permitindo assim uma abordagem holística para resolver o problema. Portanto, toda a complexidade dos dispositivos está escondida no interior das interfaces fornecidas pelos respectivos serviços.

2.2.3 *Standards* na indústria ICT

Actualmente existem cada vez mais expectativas do público para a transparência, abertura e comunicação [39]. Através do uso de *standards* são exploradas novas ferramentas que ajudam a estimular a inovação, a eficiência e a criar benefícios económicos para os consumidores finais.

Os *standards* fomentam a interoperabilidade entre produtos e serviços dentro de um mercado ajudando a promover a inovação, estimulando o crescimento do mercado e protegendo os investimentos em novas tecnologias. Eles podem catalisar a inovação, incentivando as empresas a contribuírem com a sua tecnologia inovadora, realizando actividades de definição de *standards*, partilhando a sua propriedade intelectual com outras empresas.

Por outro lado, existem casos em que os *standards* tiveram o efeito oposto, ou seja se uma determinada empresa fosse detentora de um standard e não o disponibilizasse, essa mesma empresa detinha todo o monopólio desse nicho de mercado. Um exemplo

conhecido é o da companhia de telefones AT&T que monopolizou o mercado Norte Americano de telefones, ocorrendo em 1984 a intervenção do tribunal [42].

Os *standards* ICT possuem cinco características:

- Os *standards* são uma ferramenta para promover a eficiência, interoperabilidade e inovação.
- Os processos pelos quais os *standards* ICT são criados podem variar muito e estão em constante evolução.
- Como os *standards* surgem para cumprir as diferentes necessidades dos consumidores, a sobreposição dos *standards* é comum na indústria.
- As especificações da indústria não tem que se tornar *standards* formais para criarem valor no mercado.
- Os *standards* não garantem a interoperabilidade.

Num ambiente de computação composto, onde se utiliza diferentes produtos de diferentes fabricantes, existe heterogeneidade, mas existe também algum esforço por parte dos fabricantes de software, de modo a que os diferentes programas funcionem em conjunto. O esforço dos fabricantes não é maior, porque querem vender todos os seus produtos, garantindo uma melhor interoperabilidade entre eles e não com os outros fabricantes de software, como é o caso da conhecida Microsoft. Os *standards* podem ser o melhor suporte para melhorar a interoperabilidade, quando eles são desenvolvidos e implementados através duma abordagem multifacetada que inclui:

- Método de definição dos *standards* conhecido por todos.
- Manutenção proactiva dos *standards*.
- Empenho forte e colaborativo da indústria, incluído a colaboração dum fornecedor de testes com vista a assegurar que diferentes implementações do mesmo standard, serão realmente interoperáveis.

Mesmo assim, a interoperabilidade das ICT não significa a aplicação uniforme de um único standard, devido ao uso de vários dispositivos e programas de software que usa diferentes especificações e também *standards* diferentes. Um exemplo é o software de processamento de texto que normalmente pode ler e gravar em diversos formatos de texto, os utilizadores podem utilizar assim o editor que cumpra os seus requisitos, desde que o mesmo o possibilite de usar o standard pretendido.

Capítulo 3

Requisitos e Modelo Conceptual

Este capítulo pretende apresentar ao leitor, uma descrição funcional e sucinta da visão e do objectivo da dissertação, anunciados na introdução. Serão descritos os termos básicos da dissertação, os respectivos requisitos, os elementos fundamentais, apresentada a arquitectura conceptual e, com o apoio da linguagem de modelação Unified Modeling Language (UML) mostrar-se-á uma visão funcional com os diagramas de casos de uso. Por fim irá ser mostrado o Diagrama de Entidade Relacionamento (DER) do sistema.

Por razões de clareza, importa definir um termo utilizado nesta dissertação, sendo este
Serviço: Conjunto abstracto de funcionalidades que são fornecidas.

3.1 Os Requisitos do Projecto NEMO

A infra-estrutura NEMO é suportada pelo paradigma SOA. Neste contexto, um sistema compatível NEMO é uma colecção de sistemas, de subsistemas e de dispositivos encapsulados por serviços. Isto permite um acesso directo e integrado a informações da rede, com o nível desejado de granularidade e semântica, de modo a poder tomar decisões sólidas e consistentes [28].

Para aplicar este paradigma ao nível dos dispositivos existem requisitos que se têm que cumprir, mas mantendo os mesmos perfeitamente interoperáveis. A estratégia do NEMO para lidar com esta exigência assenta em três principais pilares, nomeadamente SOA, *Web Services*, e *Standards*, como se pode observar na figura 3.1.

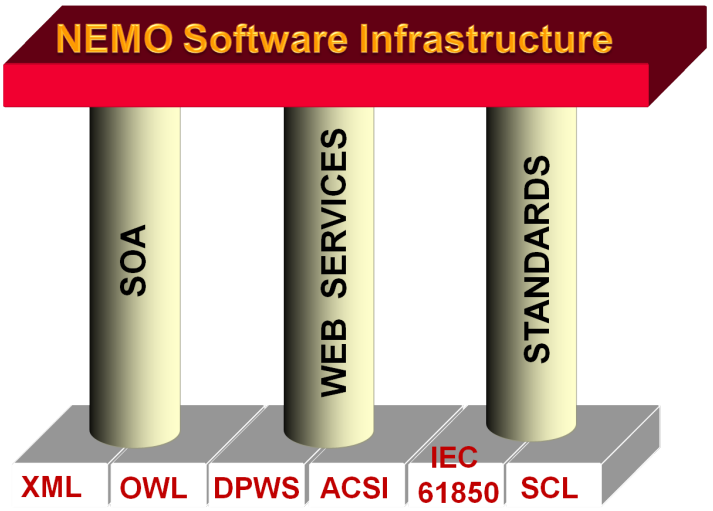

FIGURA 3.1: Os pilares do NEMO

O paradigma SOA fornece ao NEMO princípios de arquitectura e a orientação de desenvolvimento. Os *Web Services* oferecem uma maneira universal de fornecer interoperabilidade entre sistemas heterogéneos. Por sua vez, os *standards* possibilitam ao NEMO uma maneira segura de evoluir e interagir com outros sistemas no futuro.

Os requisitos do projecto NEMO podem ser divididos em duas categorias: genéricos e aqueles que fizeram decidir por um sistema baseado no paradigma SOA. Os requisitos do projecto NEMO que obedecem ao paradigma SOA são:

- **Autonomia:** Nenhuma dependência directa entre serviços, estruturalmente dissociados.
- **Interoperabilidade:** Descrição dos serviços hospedados e dos padrões de interacção.
- **Independência da plataforma:** Baseado em *standards*.
- **Encapsulamento:** Funcionalidades embutidas expostas por interfaces, escondendo detalhes desnecessários.
- **Disponibilidade/Descoberta:** Serviços publicados nos registos públicos e disponibilizados para utilização geral.

Além destes requisitos existem mais alguns importantes num sistema deste tipo, sendo estes:

- Reconhecimento e instalação de dispositivos com o mínimo de configuração manual.
- Integração de sistemas e dispositivos desconhecidos através de protocolos de comunicação baseados em *standards*.
- Composição/orquestração dinâmica de serviços.

3.2 Elementos Fundamentais

Existem dois elementos fundamentais para a infra-estrutura de software desenvolvida nesta dissertação, sendo eles: *Web Services* e *standards*. Os *Web Services* fornecem uma forma aberta e universalmente aceite para oferecer a interoperabilidade entre sistemas heterogéneos [8, 33]. O DPWS é o standard com o objectivo de influenciar o uso de *Web Services* para suportar a comunicação entre dispositivos [13]. O IEC 61850 normaliza a identificação (num grande sentido, incluindo a descrição técnica e as funcionalidades fornecidas pelo IED), da maioria dos IEDs comuns e a comunicação entre eles através da prestação de serviços genéricos, permitindo tanto a troca de dados como a execução de comandos. Nas subsecções seguintes temos uma descrição mais detalhada de cada um dos elementos fundamentais.

3.2.1 DPWS

A comissão técnica da OASIS para *Web Services* de descoberta e para o perfil de dispositivos [13], propôs o DPWS, um standard que se destina a *Web Services* com a capacidade de dar suporte à comunicação entre dispositivos. DPWS é um mediador comum entre *Web Services* e perfis de dispositivos, que define dois elementos fundamentais: o dispositivo e os serviços hospedados, como se pode observar na figura 3.2. Os dispositivos desempenham um papel importante na descoberta e nos protocolos de troca de metadados, enquanto os serviços hospedados fornecem o comportamento funcional do dispositivo e dependem do seu alojamento para a descoberta. O modo de operação mais comum de um DPWS *endpoint* consiste na descoberta dos dispositivos mais relevantes na rede, recolha da informação e do conjunto dos serviços hospedados

no dispositivo, invocação serviços no dispositivo seleccionado e a subscrição de fontes de eventos.

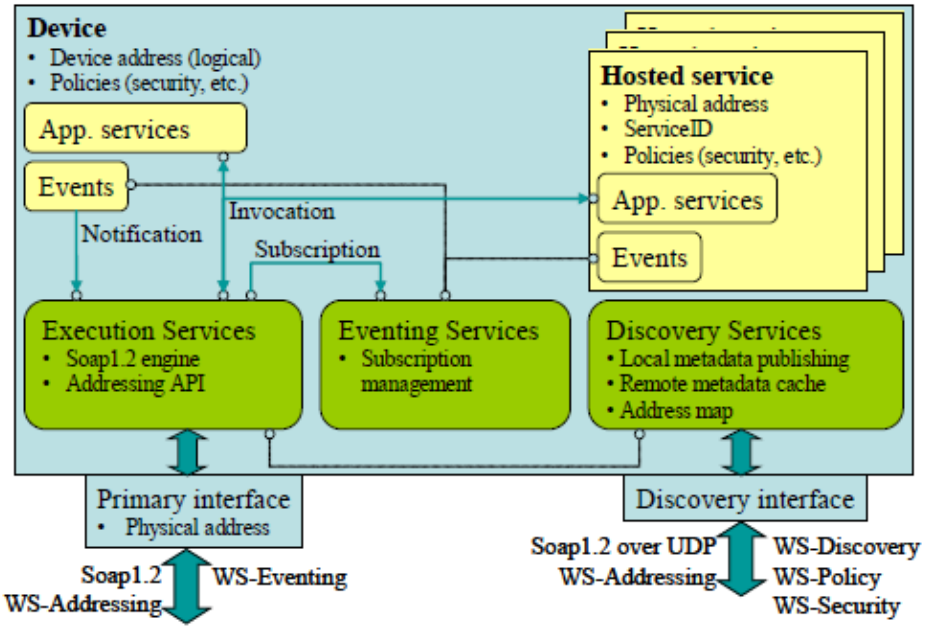

FIGURA 3.2: Visão geral da arquitectura DPWS [13].

Além dos serviços hospedados poderem ser desenvolvidos pelo utilizador final, o DPWS também especifica uma infra-estrutura de um conjunto de serviços integrados, nomeadamente:

- **Serviços de descoberta (*WS-Discovery*):** Utilizado por um dispositivo conectado a uma rede, para se anunciar a si mesmo e descobrir outros dispositivos. O dispositivo usa User Datagram Protocol (UDP) para transmitir e difundir informações e escutar as mensagens de descoberta.
- **Serviços de troca de metadados (*WS-MetadataExchange*):** Fornecem acesso dinâmico aos serviços hospedados no dispositivo e aos seus metadados, tais como as definições do WSDL e/ou as regras de validação XML.
- **Serviços de subscrição e publicação de eventos (*WS-Eventing*):** Extensão dos serviços definidos pelo fornecedor, que vai permitir que os outros dispositivos façam a subscrição de mensagens assíncronas (eventos), produzidos por um serviço definido pelo utilizador.

Ao mais alto nível, as mensagens correspondem a acções e eventos especificados pelo utilizador. Todos os protocolos de valor acrescentado descritos anteriormente, dependem do protocolo SOAP 1.2, estendido com *WS-Addressing* e *WS-Policy*. As mensagens são entregues usando os protocolos standard de transporte HTTP, Transmission Control Protocol/Internet Protocol (TCP/IP) e UDP, como se pode observar na figura 3.3.

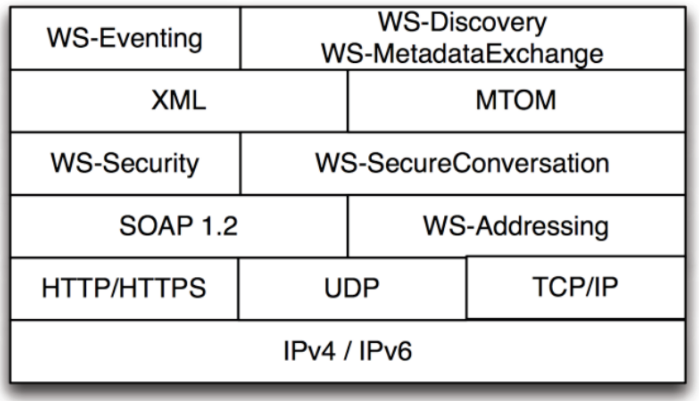

FIGURA 3.3: Pilha de protocolos DPWS [13].

A utilização de *Web Services* ao nível do dispositivo irá melhorar o funcionamento do sistema, bem como o processo de desenvolvimento [24]. Através do uso de DPWS a escalabilidade é favorecida pelo facto das comunicações despoletadas por eventos serem substancialmente mais eficientes, no que respeita à largura de banda utilizada e às complicações dos processos, do que as comunicações síncronas que estão constantemente a consultar o estado de um dispositivo. Adicionalmente, permite a perfeita integração de dispositivos na rede (por exemplo, dos vários dispositivos existentes numa fábrica), o que fará que seja uma empresa apoiada por ICT. Vale a pena salientar que o DPWS já é nativamente suportado pelos mais recentes sistemas operativos da Microsoft (Windows Vista e Windows 7).

Uma vez que o DPWS foi desenvolvido sobre *standards* baseados em Web, é facilmente integrável com outras especificações de acordo com o contexto pretendido (por exemplo, integrar DPWS e *WS-Management*, para permitir a gestão dos recursos DPWS em equipamentos de automação [8]).

O mediador DPWS foi aplicado pela primeira vez na automação industrial ao nível do dispositivo, no projecto ITEA SIRENA [22], surgindo assim uma forte base que se foi desenvolvendo para suportar outros projectos, tais como ITEA SODA [23] e FP7 IST

SOCRADES [38]. Juntamente com os proprietários de outras implementações, uma pilha de DPWS está agora disponível como um produto de código aberto [37]. DPWS fornece um pequeno conjunto eficiente de classes para uma interacção ponto a ponto entre dispositivos, que é completamente compatível com a família de especificações Web.

Apesar do objectivo final ser embutir uma aplicação compatível com DPWS directamente no hardware de cada dispositivo, é também necessário especificar soluções para integrar dispositivos já existentes na rede NEMO. Enquanto no primeiro caso um dispositivo apenas precisa de ser conectado na rede para ser descoberto, bem como para os seus serviços estarem disponíveis, o segundo caso implica o uso de uma porta de ligação DPWS para se alcançar o mesmo resultado.

Uma porta de ligação pode ser vista como uma caixa onde outros dispositivos (por exemplo: medidores de energia e relés), serão conectados caso eles não usem o mesmo protocolo de comunicação. Hoje em dia estes dispositivos comunicam normalmente usando protocolos baseados em *fieldbus*, os quais não são completamente compatíveis com uma abordagem SOA. Por isso, em vez de estarem directamente conectados à rede NEMO, estes dispositivos estarão conectados a uma porta de ligação DPWS. Esta porta de ligação DPWS será capaz de fazer a ponte entre o protocolo *fieldbus* e o DPWS em si. Como tal, um determinado dispositivo pode ser detectado quando conectado dentro da rede NEMO, desde que a porta de ligação reconheça o determinado dispositivo e crie o dispositivo DPWS equivalente, que irá abstrair esse dispositivo específico.

Este dispositivo DPWS vai-se comportar como qualquer outro dispositivo DPWS na rede. Além disso, este dispositivo é compatível com a estrutura global de projecto do NEMO. Ao nível da rede, a porta de ligação DPWS será um elemento oculto, uma vez que irá proporcionar um acesso transparente de (e para) todos os dispositivos, que a tenham que utilizar para se conectar à rede NEMO.

3.2.2 *Standards*

Os EMSs devem contar com modelos de informação bem definidos para alcançar a interoperabilidade universal dos dispositivos relacionados com energia [29]. Nas subsecções

seguintes irão ser mostrados mais detalhadamente dois dos *standards* usados nesta dissertação. Primeiro o standard IEC 61850 e, de seguida o standard de comunicação Modbus.

3.2.2.1 IEC 61850

Ao cumprir as directrizes da norma IEC 61850 para a descrição do IED e utilizando os mecanismos da norma para a interacção com o IED, a infra-estrutura de software normaliza a identificação da maioria dos dispositivos e a interacção entre eles.

O IEC 61850 é integrado na arquitectura NEMO através dos seguintes elementos particulares: o modelo de dados Abstract Communication Service Interface (ACSI), utilizado na descrição de IEDs e dos seus conjuntos específicos de características técnicas; e Substation Configuration Language (SCL), que fornece tanto a sintaxe como a semântica necessária ao modelo de dados ACSI para descrever as características técnicas dos IEDs.

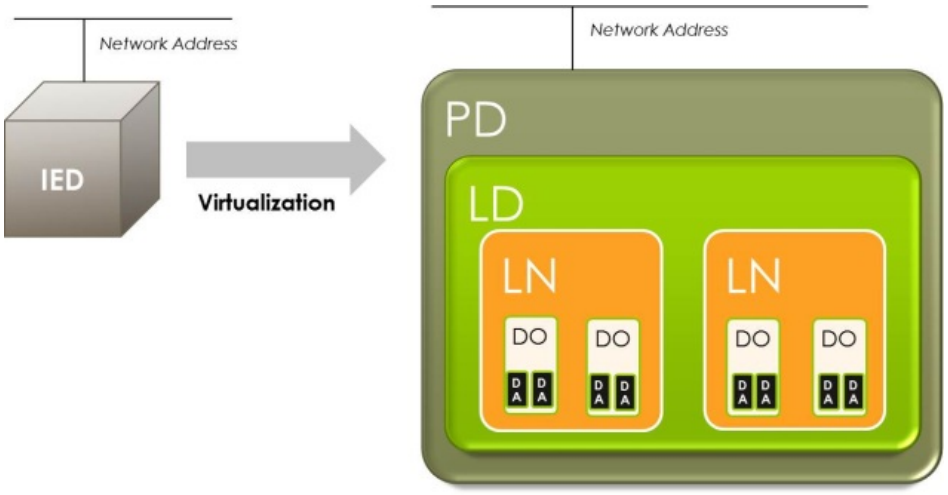

FIGURA 3.4: Virtualização de um dispositivo físico através do modelo de dados ACSI.

Conforme mostrado na figura 3.4, o IEC 61850 permite a virtualização de Physical Devices (PDs), descrevendo as funcionalidades específicas de cada dispositivo através do modelo de dados ACSI. Cada IED é decomposto em um ou mais Logical Devices (LDs), dependendo do tipo de dispositivos que o IED abrange, com cada LD herdando entidades de menor dimensão, denominadas Logical Nodes (LNs), que representam diferentes tipos de características físicas, cuja aplicação e conteúdo é descrito em [1].

Cada LN contém vários Data Objects (DOs) que fornecem informações específicas sobre o respectivo LN. Para cada DO, vários Data Attributes (DAs) fornecem informações detalhadas sobre cada característica física do dispositivo.

A informação contida em cada DA é reconhecida pelo modelo de dados ACSI e pode ser activada ou solicitada através da invocação de um serviço ACSI. No exemplo da figura 3.5, o IED tem características físicas de um disjuntor (XCBR), cuja posição é dada pelo DA stVal.

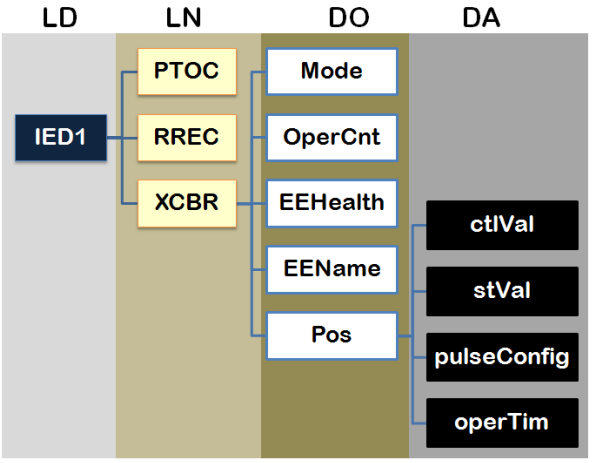

FIGURA 3.5: Exemplo de um modelo de dados ACSI.

Como mencionado anteriormente, o IEC 61850 fornece ao NEMO um modelo de dados ACSI traduzido em SCL. Logo, o modelo de dados ACSI é preenchido com conteúdos SCL para expressar as funcionalidades do IED no sistema NEMO. Um pequeno excerto do ficheiro SCL do gerador eólico pode ser visualizado na figura 3.6. Este SCL permite que o sistema NEMO obtenha os valores da potência activa trifásica (TotW), potência reactiva trifásica (TotVAr), potência aparente trifásica (TotVa), a frequência (Hz) e o factor de potência (TotPF).

O IEC 61850 define serviços standard, denominados serviços ACSI, para a interacção com o IED. As características físicas podem ser activadas e desactivadas, ou as suas informações podem ser solicitadas ou alteradas através da invocação destes serviços.

As operações do NEMO baseiam-se em dois serviços ACSI, nomeadamente *GetDataValues* e *SetDataValues*. Enquanto o primeiro é invocado para operações de monitorização, quando o conhecimento sobre o estado de uma característica física é necessário, o último permite o controlo físico de um determinado dispositivo.

```
            <LN lnType="MMXU_0" lnClass="MMXU" prefix="" inst="1">
              <DOI name="TotW" desc="ThreePhActivePower"/>
              <DOI name="TotVAr" desc="ThreePhReactivePower"/>
              <DOI name="TotVA" desc="ThreePhApparentPower"/>
              <DOI name="Hz" desc="Frequency"/>
              <DOI name="TotPF" desc="PowerFactor"/>
            </LN>
          </LDevice>
        </Server>
      </AccessPoint>
    </IED>
    <DataTypeTemplates>
     <LNodeType id="MMXU_0" lnClass="MMXU">
      <DO name="TotW" type="MV_0"/>
      <DO name="TotVAr" type="MV_0"/>
      <DO name="TotVA" type="MV_0"/>
      <DO name="Hz" type="MV_0"/>
      <DO name="TotPF" type="MV_0"/>
     </LNodeType>
     <DOType id="MV_0" cdc="MV">
      <DA name="mag" fc="MX" bType="AnalogueValue"/>
```

FIGURA 3.6: Pequeno excerto do ficheiro SCL do gerador eólico.

Os serviços fornecidos pelos IEDs compatíveis com ACSI são mapeados através de um DA utilizando uma referência específica denominada Functional Constrained Data Attribute (FCDA). Este pode ser visto como um trajecto para um dado específico e que inclui, entre outros, o LD, LN, DO e DA que caracterizam univocamente a operação física (monitorização ou controlo) a ser executada. Os atributos do elemento FCDA e respectivas descrições podem ser visualizados na tabela 3.1.

TABELA 3.1: Atributos do elemento FCDA.

Nome do atributo	**Descrição**
ldInst	O LD onde o DO se encontra
prefix	Prefixo que identifica o LN onde o DO reside, juntamente com lnInst e lnClass
lnClass	Classe do LN onde o DO se encontra
lnInst	Número da instância do LN onde o DO se encontra
doName	Nome para identificar o DO (dentro do LN)
daName	Nome do atributo
fc	Indica que todos os atributos desta restrição funcional estão seleccionados

3.2.2.2 Modbus

O Modbus é um protocolo de mensagens da camada aplicação do modelo Open Systems Interconnection (OSI), que fornece comunicação cliente/servidor entre dispositivos conectados em diferentes tipos de barramentos ou redes [30]. O Modbus foi reconhecido como um standard de comunicação em 1979 e continua a permitir que milhões de dispositivos de automação comuniquem entre si.

A interface do comunicação do Modbus é construída em torno de mensagens que são independentes da interface física usada. A estrutura das mensagens esta descrita na tabela 3.2.

Tabela 3.2: Estrutura da mensagem Modbus

Campo	**Descrição**
Endereço do dispositivo	Endereço do receptor
Código da função	Código que define o tipo de mensagem
Dados	Bloco de dados com informações adicionais
Verificação de erro	Valor de verificação para detectar erros de comunicação

O endereço do dispositivo especifica qual o dispositivo destinatário numa rede, que pode conter até 255 dispositivos. O código da função apenas indica ao servidor a acção a ser executada. O campos dos dados contém informações adicionais que o servidor utiliza para executar a função definida pelo código da função. Esses informação podem incluir elementos como endereços discretos e de registos, a quantidade de elementos a serem considerados e o número de bits actuais existentes no campo de dados. O campo de dados pode ser inexistente ou ser de comprimento zero, dependendo do código da função, neste caso, o servidor não necessita de informações adicionais.

Com esta estrutura de mensagens é possível o cliente controlar quando existe erros na mensagem enviada. Se for um envio com erros, acontecerá o descrito na figura 3.7. Para uma resposta com erros, o servidor retorna um código que é equivalente ao código da função requerida, com o seu bit mais significativo definido no nível lógico 1.

Por outro lado, se a transmissão decorrer sem erros, o servidor apenas responde para o cliente uma mensagem com o código da função original, como se pode observar na figura 3.8.

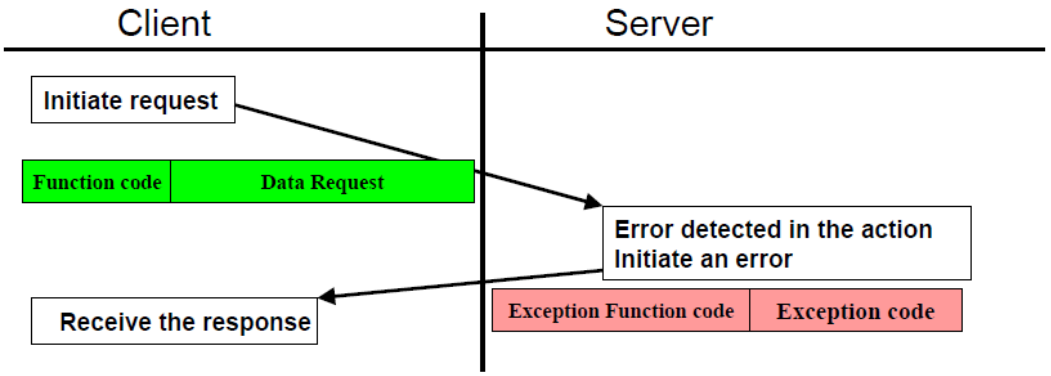

FIGURA 3.7: Transacção Modbus (resposta de excepção) [30].

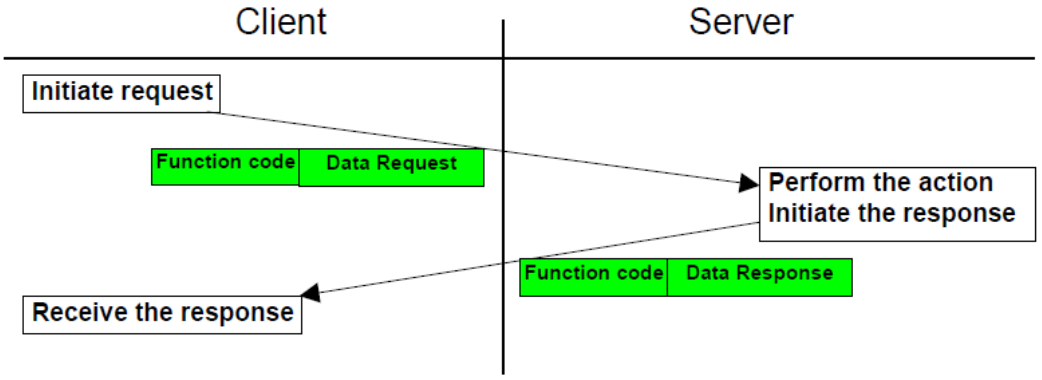

FIGURA 3.8: Transacção Modbus (resposta sem erros) [30].

Por cada função invocada, o protocolo permite a selecção individual de 65535 elementos de dados e as operação de leitura, ou a gravação desses elementos foram desenhadas para abranger elementos de dados consecutivos, até um limite de dados que depende do código da função requerida.

Todos os dados manipulados através do protocolo Modbus devem estar localizados na memória do dispositivo. Não se pode confundir o endereço físico na memória com a referência dos dados, apenas é exigido a ligação entre o endereço físico e a referência dos dados.

3.3 Visão Global

A visão global do NEMO é que as redes de dispositivos relacionados com energia podem ser operadas auxiliadas por uma infra-estrutura de software (distribuída), com

base no paradigma SOA e em *standards*. Assim, cada sistema NEMO é composto por duas redes, nomeadamente: NEMO Energy Network (NEMO EN) e NEMO Software Network (NEMO SN) (figura 3.9). A primeira rede é formada por sistemas e dispositivos de produção, distribuição e consumo de energia, enquanto a rede de software é usada para monitorizar e controlar a rede de energia.

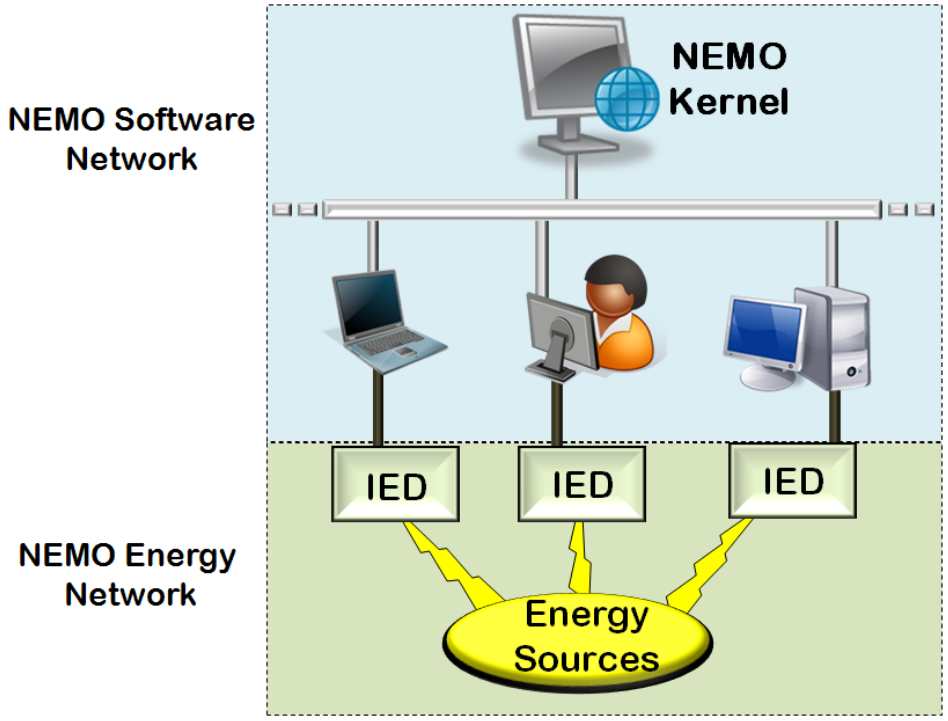

FIGURA 3.9: Visão global de um sistema NEMO.

As redes de energia consideradas pelo NEMO podem usar ambos os dispositivos novos e *legacy*, classificados como Intelligent Electronic Devices (IEDs) . Eles são obrigados a ter um mínimo de "inteligência" para serem virtualizados num dado sistema NEMO. Por outras palavras, eles têm de fornecer um canal de comunicação com o software, que será usado para uma comunicação baseada em *standards* com a NEMO SN.

A figura 3.10 oferece uma análise mais atenta sobre os principais componentes da NEMO SN, mostrando cinco componentes, nomeadamente NEMO-Kernel (NEMO-K), NEMO-Api (NEMO-A), NEMO-Bus (NEMO-B), NEMO-Connector (NEMO-C) e IEDs.

Os IEDs estão no nível mais próximo a respeito de dispositivos (na figura 3.10: pilha de combustível, painéis fotovoltaicos, gerador eólico, medidores de energia e uma porta

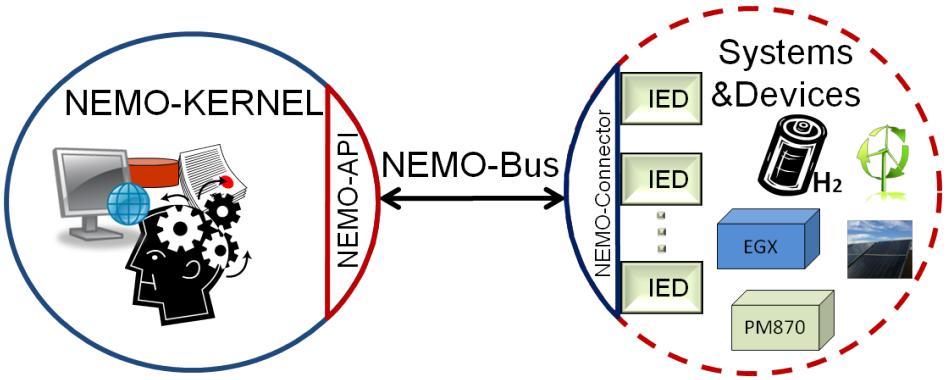

FIGURA 3.10: Os principais componentes da rede de software NEMO.

de ligação *Ethernet*) e através deles, os comandos são enviados para os dispositivos ou os dados são recolhidos.

A estrutura de um IED do sistema NEMO está representada na figura 3.11. Ele contém a unidade de processamento (fornecida pelo fabricante), que garante a comunicação com o dispositivo físico em si, ou seja, a própria troca de dados de (e para) o dispositivo (fluxo de dados).

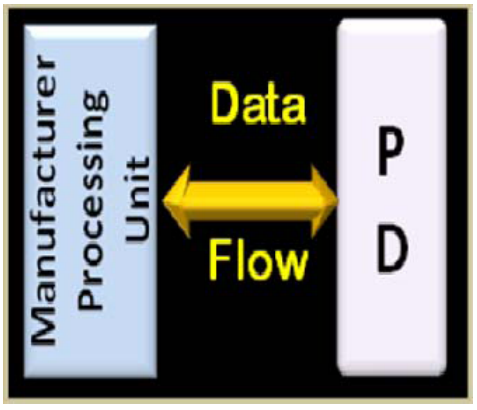

FIGURA 3.11: Estrutura interna dos IEDs.

No lado oposto, temos o NEMO-K, que é o orquestrador do funcionamento da sua instância do sistema NEMO. O NEMO-K gerencia os serviços que podem ser fornecidos pela sua instância para o resto do mundo. Esses serviços são denominados NEMO External Service (NEMO ES), implementados através de *Web Services*.

O NEMO-B suporta a interacção entre o NEMO-K e todos os dispositivos encontrados na rede. Ele representa o canal de comunicação através do qual os ficheiros necessários para o funcionamento do NEMO irão fluir para trás e para a frente. Quando os

comandos são enviados para o dispositivo ou quando os dados operacionais devem ser adquiridos.

O NEMO-C é uma espécie de "embaixador do sistema NEMO", uma vez que é responsável pela própria integração e virtualização de IEDs numa determinada instância do sistema NEMO. O NEMO-C fornece um canal de comunicação através de DPWS para a troca do SCL entre todos os IEDs individuais e o seu respectivo NEMO-K.

Para concluir esta subsecção, é oportuno relembrar que cada par de redes NEMO (software e energia) constituem um exemplo do sistema NEMO. Cada exemplo oferece os seus serviços para o mundo exterior a fim de permitir, por exemplo, o acompanhamento, e/ou acções de controlo a partir de um ponto de vista superior.

3.4 Visão Estrutural

A arquitectura do NEMO é composta por dois elementos principais: o NEMO-K e o NEMO-C. Como mencionado anteriormente na subsecção 3.3, o NEMO-C é responsável pela integração dos IEDs numa determinada instância do sistema NEMO, permitindo que o NEMO-K execute operações nos IEDs virtualizados pelo NEMO-C.

3.4.1 NEMO-C

O NEMO-C está estruturado em duas camadas, nomeadamente: *Services Layer* e *Communication Layer* (figura 3.12). A primeira é dividida nos seguintes componentes: *File Transfer*, *NEMO.IS*, *NEMO.IS.to.NonACSI* e *NEMO.IS.to.ACSI*. A última camada é estruturada nos seguintes componentes: *NonACSI.to.IED*, *ACSI.Wrapper* e *Communicator*. A figura 3.12 é completada por um canal directo de comunicação para os IEDs compatíveis com o standard IEC 61850, através do protocolo ACSI.

No interior da *Services Layer*, o componente *File Transfer* é responsável pela transferência de ficheiros entre o NEMO-K e o NEMO-C. O componente *NEMO.IS* processa os pedidos provenientes do NEMO-K que invocam a execução dos NEMO Internal Services (NEMO ISs). *NEMO.IS* é na verdade, um servidor DPWS que oferece quatro serviços Web, nomeadamente:

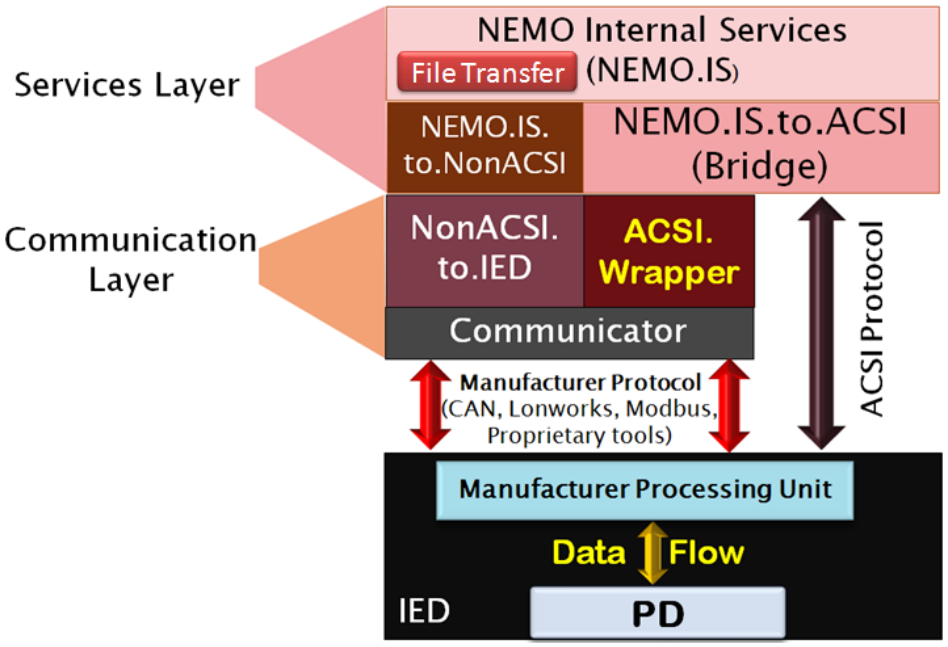

FIGURA 3.12: Arquitectura do NEMO-C.

- *GetIEC(FCDA)* e *PutIEC(FCDA, new_value)* para lidar com os serviços compatíveis com ACSI.
- *GetNonIEC(ServiceID)* e *PutNonIEC(ServiceID, new_value)* para lidar com os serviços não compatíveis com ACSI.

Enquanto os serviços *GetIEC()* e *PutIEC()* irão desencadear o componente *NEMO.IS.to.ACSI*, tanto o *GetNonIEC()* como o *PutNonIEC()* irão accionar o componente *NEMO.IS.to.NonACSI*. O *NEMO.IS.to.NonACSI* converte o NEMO IS para o serviço específico de um determinado dispositivo físico e, através do componente *NonACSI.to.IED*, esta operação é realizada através do *Communicator*, que interage com o IED utilizando o protocolo ou ferramenta de comunicação adequada (por exemplo, LonWorks, CAN, Modbus, ferramentas proprietárias, etc...).

A operação do *NEMO.IS.to.ACSI* depende do tipo do IED que está a ser utilizado. Para IEDs compatíveis com o standard IEC 61850, os serviços requeridos são convertidos pelo *NEMO.IS.to.ACSI* e enviados directamente para o IED através do protocolo ACSI. Para IEDs não compatíveis com o standard, o *ACSI.Wrapper* lida com a última fase do processo, convertendo os pedidos para um serviço específico que será invocado no *Communicator*.

3.4.2 NEMO-K

O NEMO-K é constituído por vários componentes (figura 3.13), nomeadamente:

- ***Artificial Intelligence*:** Componente onde está toda a inteligência artificial do sistema NEMO, responsável por orquestrar o sistema de modo a atingir o máximo de eficiência energética, mantendo sempre um mínimo de conforto possível.
- ***Monitor*:** Componente que vai monitorizar todos os dispositivos relacionados com energia aos quais o NEMO-K se encontra conectado.
- **NEMO ES:** Componente responsável pelas comunicações com as entidades externas que têm permissão para aceder a esta instância NEMO.
- **NEMO *Databases*:** Este componente é composto pelas três bases de dados que compõem o NEMO-K, nomeadamente:
  - ***IEDs*:** Guarda a informação referente aos IEDs que estão conectados na NEMO SN.
  - ***Users*:** Guarda a informação referente às entidades externas que acedem ao NEMO-K através dos NEMO ES.
  - ***Services*:** Guarda a informação sobre os NEMO IS disponíveis.
- **NEMO-Viewer (NEMO-V):** Interface que permite a visualização e a interacção com todos os IEDs conectados à NEMO SN, é constituído pelos seguintes componentes:
  - **NEMO IS:** Componente responsável pelas comunicações internas na rede NEMO.
  - ***File Manager*:** Componente responsável pela gestão e armazenamento dos ficheiros recebidos através dos NEMO IS.
  - ***Setup/Configuration*:** Componente responsável por realizar a configuração e instalação inicial da instância do NEMO-K.
  - ***Discovery*:** Componente responsável por descobrir quando se conectam novos dispositivos na NEMO SN, despoletando a troca de ficheiros inicial.

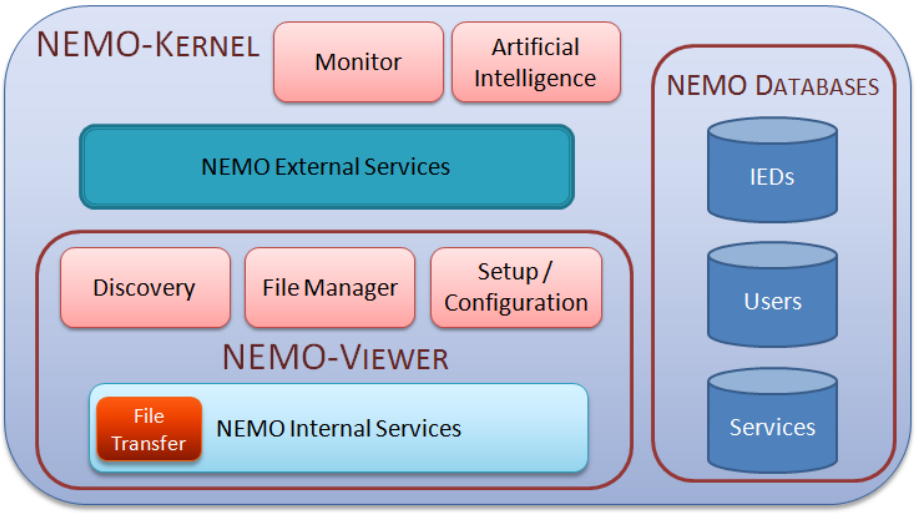

FIGURA 3.13: Arquitectura do NEMO-K.

É importante referir que o NEMO-K não foi completamente implementado nesta dissertação, devido a ser um trabalho conjunto. Como tal, o componente do NEMO-K implementado nesta dissertação foi o NEMO-V.

3.5 Visão Funcional

A vista funcional da infra-estrutura implementada é expressa a partir de diagramas de caso de uso da UML, demonstrando as funcionalidades que os utilizadores devem esperar que um sistema disponibilize. No caso de um sistema NEMO serão evidenciados dois casos de uso: um para o NEMO-V e outro para o NEMO-C.

O diagrama de casos de uso do NEMO-V é apresentado na figura 3.14 e inclui dois actores: o Utilizador e o NEMO-C. Para o NEMO-C, o NEMO-V tem a funcionalidade **Disponibilizar Ficheiros** que permite ao NEMO-V saber qual o tipo e os serviços fornecidos pelo IED. Para o Utilizador, o NEMO-V providencia as seguintes funcionalidades:

- **Realizar leitura de valores:** Consiste em realizar a leitura de valores num determinado IED.
- **Realizar actuações:** Consiste em actuar num determinado IED

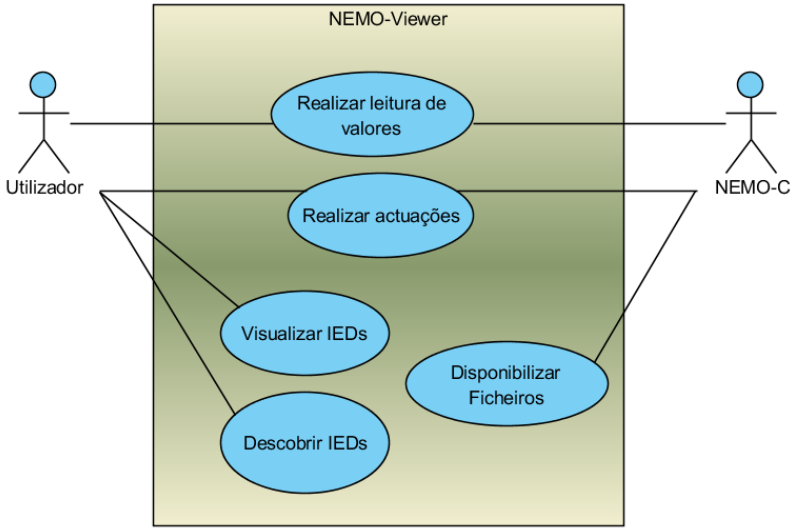

FIGURA 3.14: Casos de uso do NEMO-V.

- **Visualizar IEDs:** Permite visualizar todos os IEDs conectados à NEMO SN.
- **Descobrir IEDs:** Permite ao utilizador activar o mecanismo de descoberta.

O diagrama de casos de uso do NEMO-C é apresentado na figura 3.15 e inclui dois actores: NEMO-V e o IED. Para o NEMO-V, o NEMO-C oferece a funcionalidade **Disponibilizar ficheiros** a qual permite realizar a transferência dos ficheiros necessários para o seu funcionamento. Para o IED, o NEMO-C providencia as seguintes funcionalidades:

- **Realizar actuações:** Actua no IED para realizar os pedidos do NEMO-V.
- **Realizar leitura de valores:** Executa a leitura dos valores no IED ao qual o NEMO-C se encontra acoplado.

3.6 DER

Como referido anteriormente, o NEMO-K contém três bases de dados. Apesar de não terem sido implementadas nesta dissertação, é importante referir como elas foram

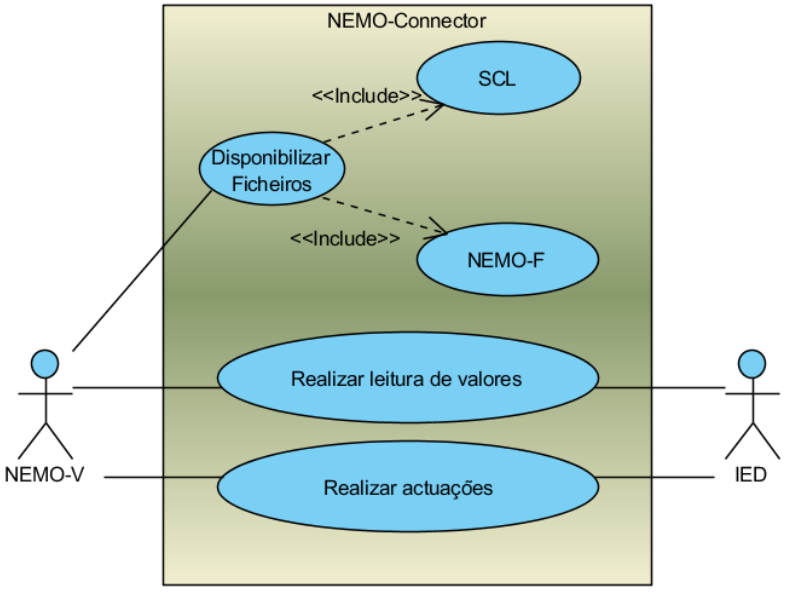

FIGURA 3.15: Casos de uso do NEMO-C.

projectadas, uma vez que vão guardar a informação proveniente dos ficheiros SCL e NEMO-File (NEMO-F).

A figura 3.16 ilustra um excerto do Diagrama de Entidade Relacionamento (DER) da tabela criada para guardar as informações relacionados com o IEDs que estão conectados na NEMO SN. O DER completo pode ser visualizado no apêndice A.

Na tabela **NemoFile** encontra-se os dados provenientes do NEMO-F do NEMO-C. Nesta tabela apenas se visualiza o cabeçalho do ficheiro XML recebido, mas existem outras tabelas com toda a informação referente ao NEMO-F.

Na tabela **SCL** encontra-se os dados provenientes do ficheiro SCL do NEMO-C. Os dados do ficheiro SCL incluem, entre outros, o cabeçalho XML e o cabeçalho do ficheiro SCL.

Na tabela **IED** encontra-se os dados que caracterizam os IEDs que o NEMO-C virtualiza. O tipo e o nome do IED estão presentes na tabela, assim como o fabricante e a versão do IED.

Na tabela **DEVICE** encontra-se os dados que caracterizam os IEDs mas que não obedecem ao modelo de dados ACSI. Esses dados são: o número de série do dispositivo, o modelo, o nome através do qual o IED é designado e o tipo do dispositivo.

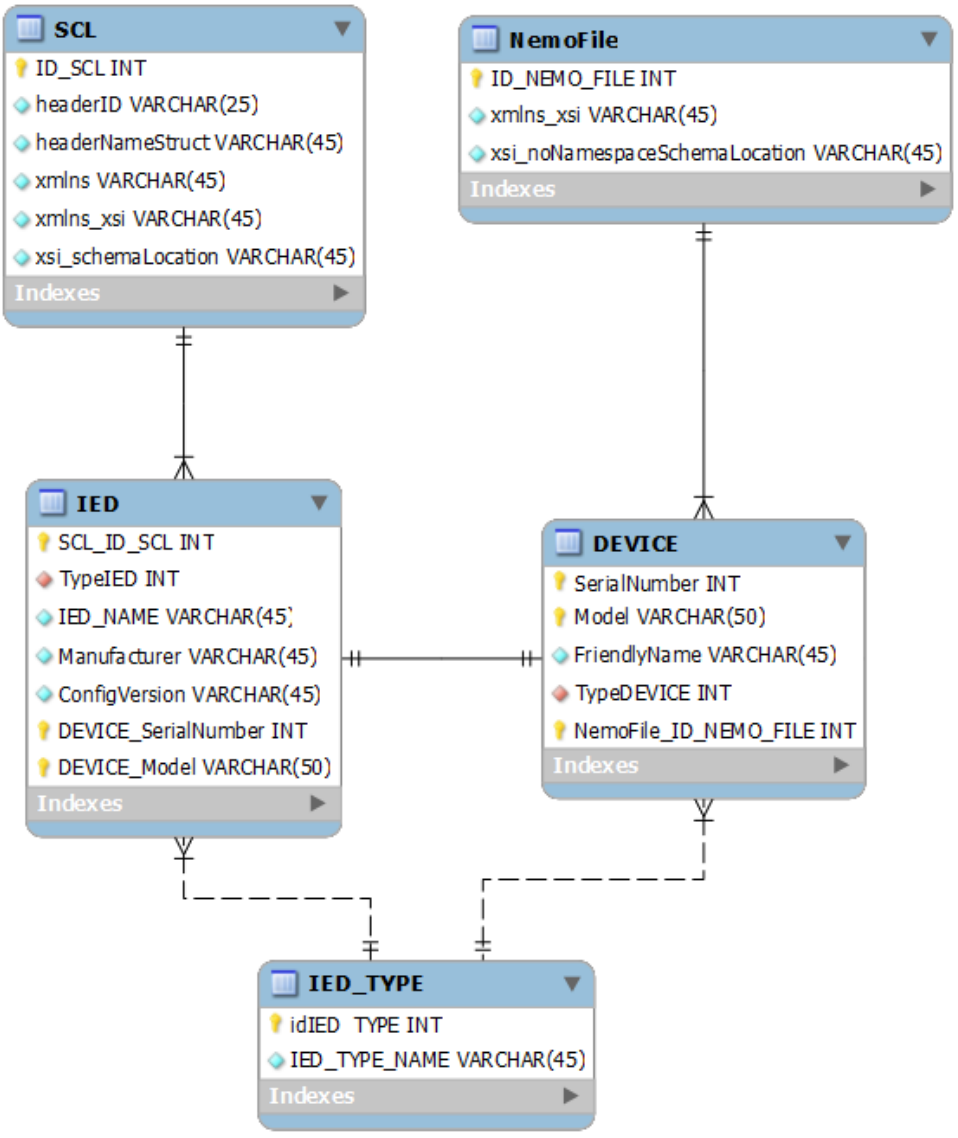

FIGURA 3.16: Excerto do DER da base de dados IEDs.

Finalmente, de modo a restringir o tipo de IEDs e dos dispositivos foi criada a tabela **IED_TYPE**, que contém os tipos implementados *Distribution*, *Consumption* e *Production*.

Capítulo 4

Implementação

Este capítulo apresenta ao leitor a implementação usada na infra-estrutura de software. São descritas as tecnologias usadas na dissertação, a configuração experimental e a visão estática e dinâmica através do apoio da linguagem UML. Por fim irá ser mostrado um exemplo de validação da infra-estrutura de software.

4.1 Escolhas Tecnológicas

Na implementação da infra-estrutura de software foram usadas várias tecnologias, as quais estão descritas na tabela 4.1.

TABELA 4.1: Escolhas Tecnológicas usadas nesta dissertação.

Escolhas Tecnológicas	**Descrição**	**Porque foi usada?**
Java	A linguagem de programação Java foi desenvolvida para ir de encontro aos desafios do desenvolvimento de aplicação num contexto heterogéneo, numa rede de ambientes distribuídos [19].	Houve uma restrição quanto ao uso da linguagem de programação. Uma das tecnologias usadas, DPWS, apenas era disponibilizada em duas linguagens: C e Java. Foi escolhido Java porque permite uma programação através do uso de classes, algo que o C não permite.
Netbeans	É um ambiente de desenvolvimento integrado de código aberto e uma plataforma de aplicações que permite que o programador crie rapidamente aplicações Web, para empresas e também aplicações móveis utilizando a plataforma Java [32].	O NetBeans permite aos programadores rapidamente criar aplicações Web, para empresas e também aplicações móveis utilizando a plataforma Java, tais como PHP, JavaScript, Ajax e C/C++ [32].
Visual Paradigm for UML	Ferramenta utilizada para o desenvolvimento do diagrama entidade relacionamento e dos diagramas UML, nomeadamente, os diagramas de casos de uso, de classe e de sequência [44].	Foi utilizada para o desenvolvimento dos diagramas UML presentes nesta dissertação.

4.2 Configuração Experimental

O cenário experimental usado para avaliar e analisar os resultados produzidos pelo projecto NEMO abrange as principais etapas do processo de energia (figura 4.1), nomeadamente: produção (baseada estritamente em fontes de energia renovável), distribuição e consumo.

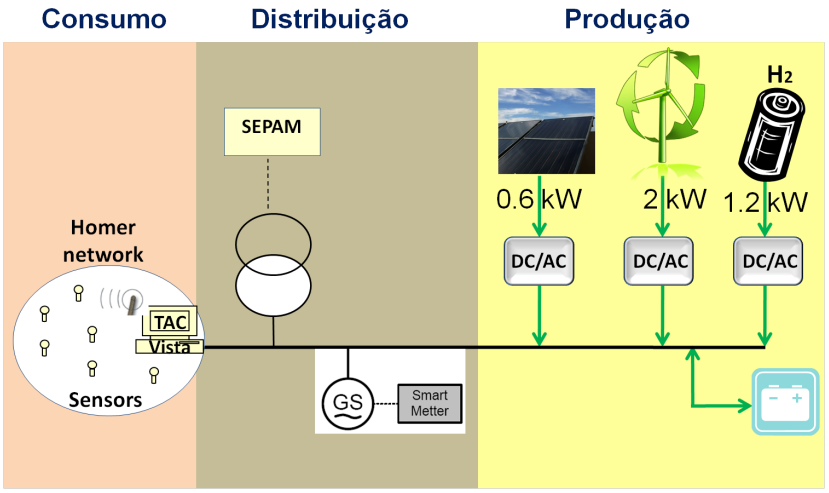

FIGURA 4.1: A configuração experimental do NEMO.

Nesta dissertação as três etapas foram implementadas. Na parte de produção foram integradas três fontes de energia renovável, nomeadamente:

- Um conjunto de painéis fotovoltaicos, com uma potência total instalada de 0,6 kW.
- Uma turbina eólica com uma potência instalada de 2 kW.
- Uma pilha de combustível com uma potência instalada de 1,2 kW.

Na etapa da distribuição foi integrado um relé digital para protecção eléctrica. O acesso centralizado a estes equipamentos vai permitir uma gestão rápida e eficiente da rede distribuída de energia, permitindo a sua reconfiguração em caso de falha de um elemento produtor de energia. Na etapa do consumo medidores inteligentes de energia foram integrados de modo a permitir a monitorização. Como referido na secção 3.3, o NEMO-K não foi completamente implementado, ficando assim com a infra-estrutura de software da figura 4.2.

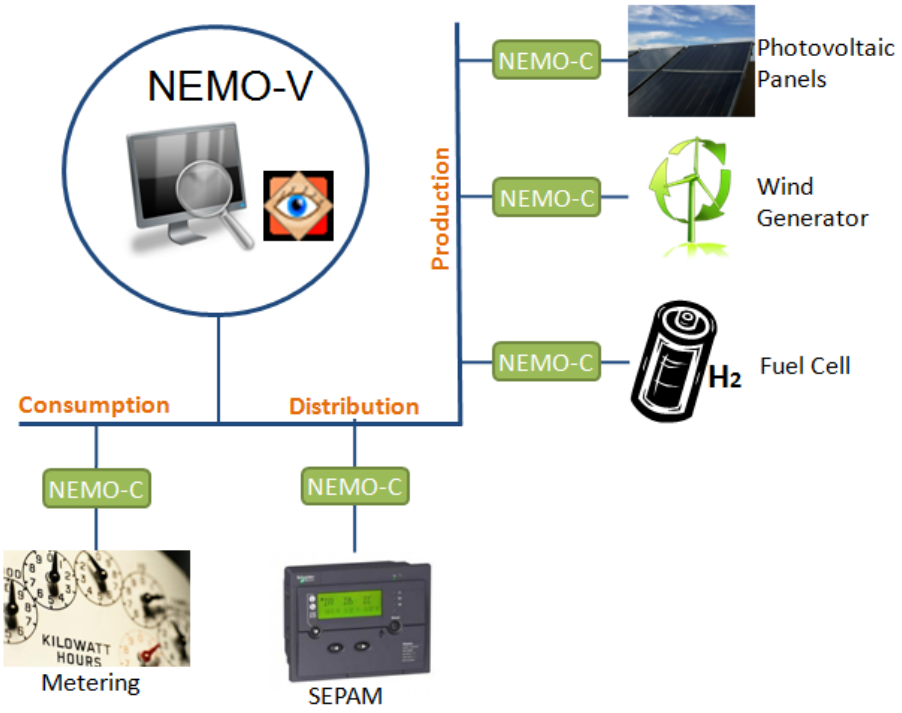

FIGURA 4.2: A configuração experimental da infra-estrutura implementada.

A produção de energia renovável está a operar de forma eficiente. O custo de energia irá depender da procura, do custo do combustível da pilha (hidrogénio), bem como da electricidade proveniente do fornecedor. Nesta configuração experimental irá ser utilizada energia renovável gratuita (energia fotovoltaica e eólica) sempre que possível.

4.3 NEMO-F

Devido a não ser possível executar o mapeamento para ACSI de alguns serviços disponíveis nos IEDs, foi necessário o desenvolvimento do NEMO-File (NEMO-F).

O NEMO-F é composto por uma lista de IEDs (*Devices*) que são virtualizados pelo NEMO-C, ao qual estão conectados. Cada IED (*Device*) contém os seguintes campos:

- ***IED61850*:** Indica se o IED é compatível com o ACSI.
- ***FriendlyName*:** Indica o nome através do qual o IED é designado.
- ***SerialNumber*:** Número de série do IED.

- ***Model*:** O modelo do IED.
- ***TypeIED*:** Indica qual o tipo de IED, se é de consumo, produção ou distribuição: *Consumption*, *Production* ou *Distribution*.
- **Services:** Contém a lista de serviços não compatíveis com o standard que o IED oferece.

A estrutura *Services*, do componente *Device*, é composta por uma lista de Serviços (*Service*), que contêm as informações necessárias para o NEMO-K saber o que esperar do serviço que vai invocar. O *Service* contém os seguintes campos:

- ***ServiceID*:** A identificação do serviço.
- ***Type*:** O tipo do serviço, se é de leitura, escrita, ou ambos: *Get*, *Put* ou *GetPut*.
- ***InputUnits*:** As unidades em que o valor é inserido.
- ***OutputUnits*:** As unidades em que o valor é devolvido.

O ficheiro foi definido em XML, que por ser uma linguagem de marcação é facilmente editável, tanto por máquinas como por humanos e é também um meio de excelência para efectuar a integração de dados entre sistemas heterogéneos. Um excerto do NEMO-F da pilha de combustível é mostrado na figura 4.3, onde se pode observar todos os campos descritos anteriormente.

Neste pequeno excerto de um NEMO-F pode-se observar os dois IEDs que que vão ser virtualizados pelo NEMO-C: A pilha de combustivel (*Nexa Fuel Cell*) e o inversor da mesma (*Hydro Boy*). O primeiro contém apenas um serviço, o qual permite saber o estado do IED, enquanto que o segundo IED contém dois serviços: *SerialNumber*, que permite obter o número de série do IED e *EnOp*, que permite ligar/desligar o inversor, ou saber se ele se encontra ligado ou desligado.

```
<NemoFile>
    <Devices>
        <Device>
            <IED61850>false</IED61850>
            <FriendlyName>Nexa Fuel Cell</FriendlyName>
            <SerialNumber>01801</SerialNumber>
            <Model>310-002702</Model>
            <TypeIED>Production</TypeIED>
            <Services>
                <Service>
                    <ServiceID>Status</ServiceID>
                    <Type>Get</Type>
                    <InputUnits>n/A</InputUnits>
                    <OutputUnits>n/A</OutputUnits>
                </Service>
            </Services>
        </Device>
        <Device>
            <IED61850>false</IED61850>
            <FriendlyName>Hydro Boy</FriendlyName>
            <SerialNumber>1600001340</SerialNumber>
            <Model>HB1124</Model>
            <TypeIED>Production</TypeIED>
            <Services>
                <Service>
                    <ServiceID>SerialNumber</ServiceID>
                    <Type>Get</Type>
                    <InputUnits>n/A</InputUnits>
                    <OutputUnits>n/A</OutputUnits>
                </Service>
                <Service>
                    <ServiceID>EnOp</ServiceID>
                    <Type>GetPut</Type>
                    <InputUnits>n/A</InputUnits>
                    <OutputUnits>n/A</OutputUnits>
                </Service>
            </Services>
        </Device>
    </Devices>
</NemoFile>
```

FIGURA 4.3: Excerto do NEMO-F da pilha de combustível.

4.4 Implementação dos diversos NEMO-Cs

Na infra-estrutura de software implementada foram integrados cinco NEMO-Cs através de sete IEDs diferentes. Os IEDs virtualizados foram, nomeadamente: pilha de combustível, um medidor de energia através de um conversor RS-485/*Ethernet* PD8, um medidor de energia através de um conversor RS-485/RS-232 e um Ethernet Gateway (EGX), que irão ser descritos em pormenor nas subsecções seguintes.

4.4.1 NEMO-C da pilha de combustível

O NEMO-C da pilha de combustível interage com dois IEDs: a pilha de combustível em si e o inversor da mesma, deste modo é possível actuar na pilha, mas também controlar a energia produzida entregue à rede. A comunicação com a pilha de combustível e com o inversor é efectuada por RS-232, mas através de dois métodos diferentes. Para a comunicação por RS-232 foi necessária um biblioteca denominada RXTX [35]. O fabricante do inversor disponibilizou uma biblioteca, denominada *YasdiMaster*, de modo a facilitar o acesso. Deste modo o NEMO-C da pilha de combustível ficou implementado como mostra a figura 4.4.

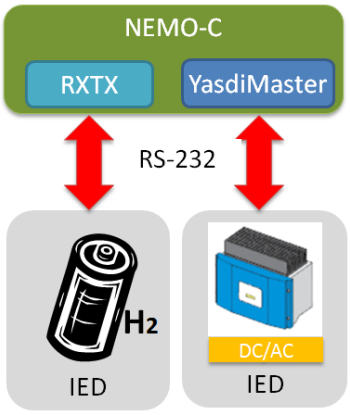

Figura 4.4: NEMO-C da pilha de combustível.

Este NEMO-C é específico para a pilha de combustível. A biblioteca disponibilizada pelo fabricante tornou-se muito útil para a implementação deste NEMO-C porque permitiu a interacção com o inversor de um modo "standard".

4.4.2 NEMO-C do conversor RS-485/*Ethernet* PD8

O NEMO-C do PD8 permite interagir com qualquer IED, através de Modbus, que esteja ligado por RS-485 ao PD8. A implementação foi executada através de um biblioteca disponibilizada para Java, denominada Jamod [25]. O IED implementado é composto por um medidor de energia (UPT210) que se encontra conectado aos painéis fotovoltaicos, como se pode observar na figura 4.5.

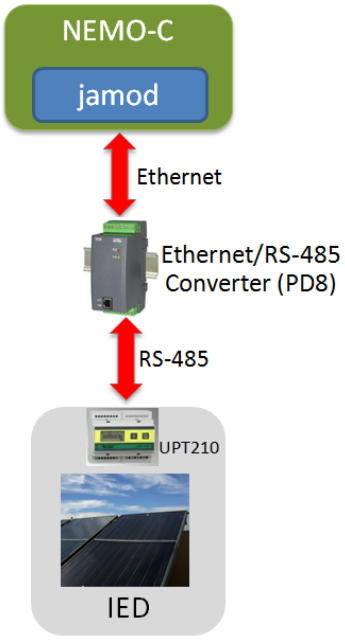

FIGURA 4.5: NEMO-C conectado ao PD8.

No entanto o PD8 tem a limitação de apenas conseguir comunicar com um dispositivo ao mesmo tempo, apesar de permitir que vários estejam conectados. Devido a este facto, o NEMO-C da subsecção seguinte foi implementado.

4.4.3 NEMO-C do conversor RS-485/RS-232

A implementação do NEMO-C do RS-485/RS-232 é o mais utilizado nesta dissertação. Ele permite comunicar com qualquer IED que comunique por Recommended Standard 485 (RS-485) e Modbus. Foi também implementado através da biblioteca jamod descrita anteriormente. Na figura 4.6 pode-se observar uma das configurações possíveis, neste caso o NEMO-C virtualiza o IED do gerador eólico.

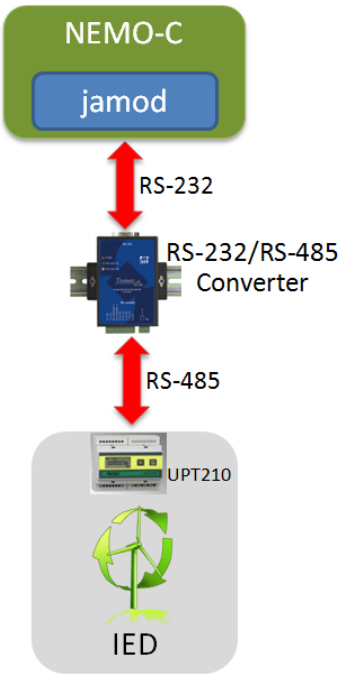

FIGURA 4.6: NEMO-C conectado através do conversor RS-485/RS-232 ao gerador eólico.

Este NEMO-C demonstra as funcionalidades *Plug and Play*. Ao alterar o NEMO-F e o ficheiro SCL podemos conectar este NEMO-C tanto a IEDs relacionados com a produção (gerador eólico, painéis fotovoltaicos), distribuição (relé digital (SEPAM)) ou mesmo da parte do consumo, através de medidores de energia ligados aos dispositivos e/ou sistemas que queremos monitorizar.

4.4.4 NEMO-C do EGX

O NEMO-C conectado ao EGX permite conectar vários dispositivos sem a limitação do PD8 referida anteriormente. O EGX é um protótipo disponibilizado pela Schneider Electric para o projecto NEMO. A ligação com o EGX é realizada através de *Web Services* que permitem a comunicação com medidores de energia (PM870 e o PM9C) (figura 4.7).

A conexão com o EGX permite a total abstracção do protocolo que o EGX utiliza para comunicar com os dispositivos conectados, devido à comunicação ser efectuada apenas por *Web Services*.

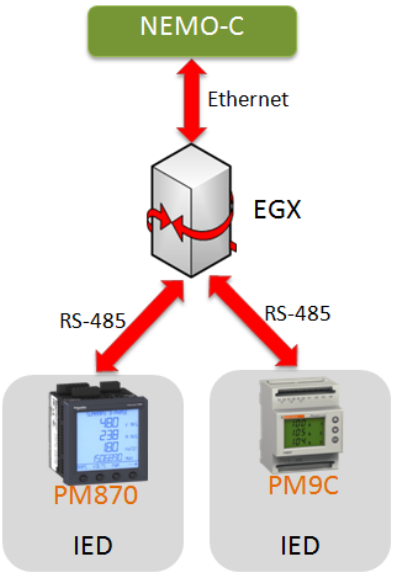

FIGURA 4.7: NEMO-C conectado ao EGX.

4.5 Visão Estática

A vista estática da infra-estrutura de software é expressa a partir dos diagramas de classe da linguagem UML. Estes diagramas apresentam as respectivas classes, as suas componentes e a maneira como as classes estão relacionadas. Apresenta-se como um esquemático da organização do sistema, mas não exibe o comportamento das classes durante o funcionamento do sistema. Os diagramas de classes do NEMO-C serão apresentados de acordo com a arquitectura apresentada na subsecção 3.4.1.

4.5.1 *Services Layer*

Como referido na subsecção 3.4.1, a *Services Layer* é dividida nos seguintes componentes: *File Transfer*, *NEMO.IS*, *NEMO.IS.to.NonACSI* e *NEMO.IS.to.ACSI*.

4.5.1.1 *NEMO.IS*

Como referido anteriormente, o componente *NEMO.IS* é um servidor DPWS (figura 4.8). É responsável por criar os perfis dos modelos dos dispositivos (*createDevice*) e alojar os serviços que permitem a comunicação com o NEMO-C.

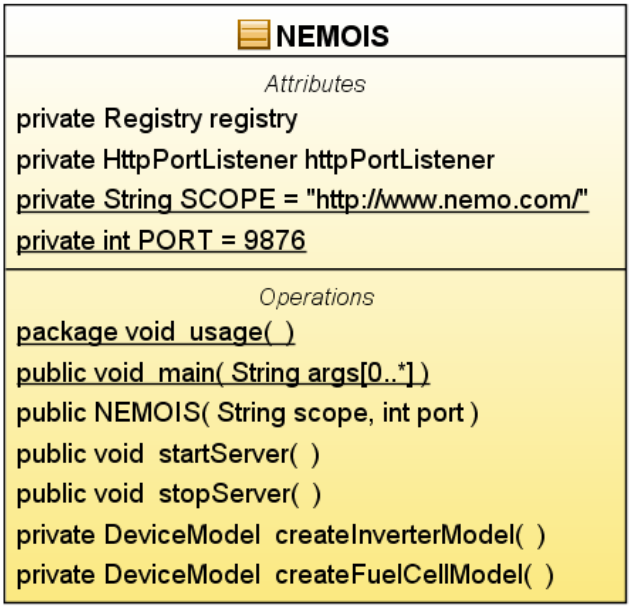

FIGURA 4.8: Diagrama de classe do componente *NEMO.IS*.

Os serviços DPWS *OperationsService* (figura 4.9) permitem a invocação dos NEMO ISs, enquanto o serviço DPWS *File Transfer* (figura 4.10) permite a transferência dos ficheiros SCL e NEMO-F. O ficheiro que descreve o serviço DPWS *File Transfer*, é apresentado no apêndice B.

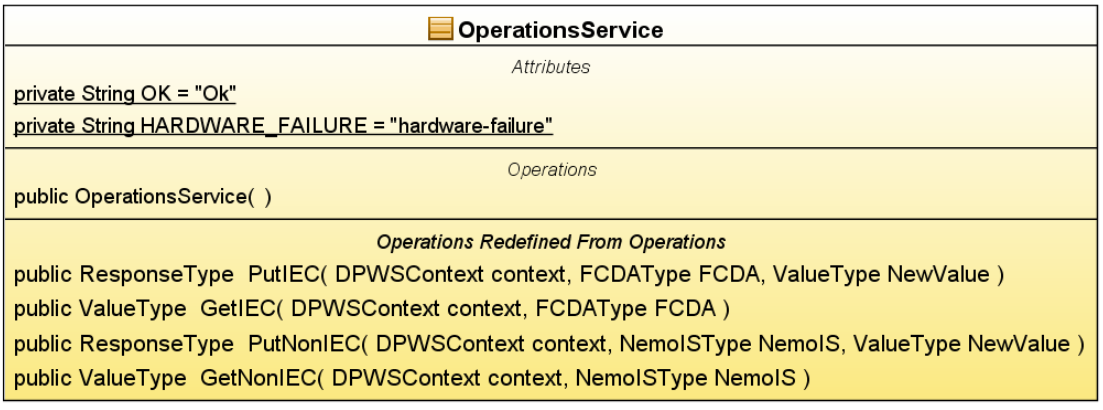

FIGURA 4.9: Diagrama de classe dos *OperationsService* embutido no componente *NEMO.IS*.

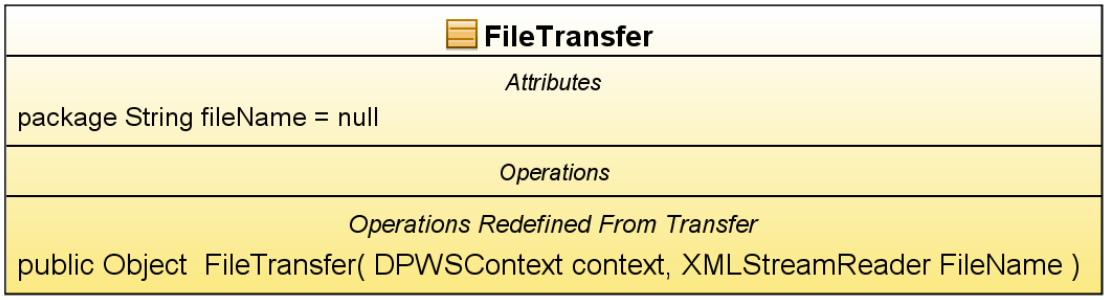

FIGURA 4.10: Diagrama de classe do *File Transfer* embutidos no componente *NEMO.IS*.

4.5.1.2 *NEMO.IS.to.NonACSI*

Este componente vai ser responsável por receber os serviços encaminhados pelo componente *NEMO.IS* e pela confirmação se é possível executar o serviço requerido, através da leitura do NEMO-F (figura 4.11). Quando as funções *get* ou *put* são invocadas e se for possível executar o serviço, o serviço vai ser encaminhado para o componente *NonACSI.to.IED* da *Communication Layer*.

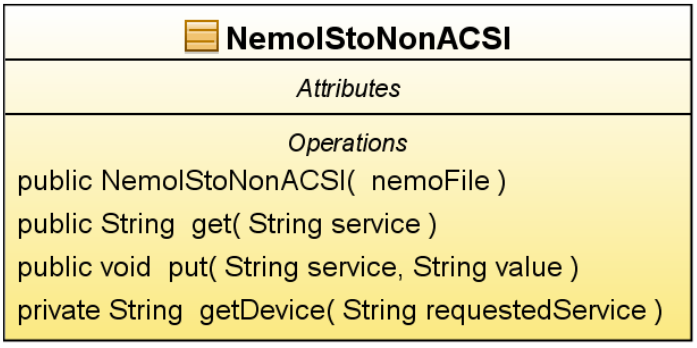

FIGURA 4.11: Diagrama de classe do componente *NEMO.IS.to.NonACSI*.

4.5.1.3 *NEMO.IS.to.ACSI*

Este componente recebe os serviços encaminhados pelo componente *NEMO.IS* e realiza a confirmação da exequibilidade do serviço requerido, através da leitura do ficheiro SCL (figura 4.11). Quando as funções *get* ou *put* são invocadas e se o serviço for exequível, o serviço vai ser encaminhado directamente para o IED caso seja compatível o standard IEC 61850, ou para o componente *ACSI.Wrapper* da *Communication Layer* caso não seja compatível.

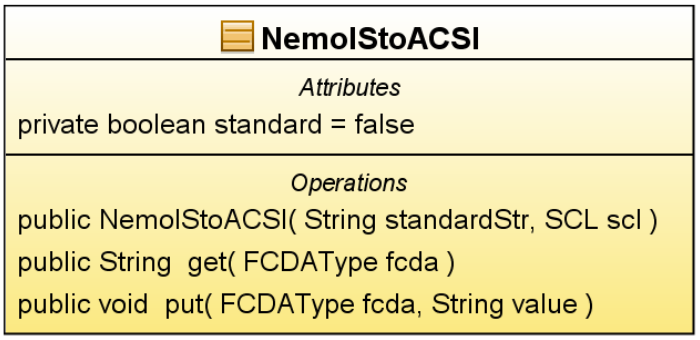

FIGURA 4.12: Diagrama de classe do componente *NEMO.IS.to.ACSI*.

4.5.2 *Communication Layer*

A *Communication Layer* é estruturada nos seguintes componentes: *NonACSI.to.IED*, *ACSI.Wrapper* e *Communicator*, descrito nas subsecções seguintes.

4.5.2.1 *NonACSI.to.IED*

O componente *NonACSI.to.IED* é responsável por oferecer serviços ao componente da camada *Services Layer*, *NEMO.IS.to.NonACSI*, de modo a que este possa invocar os serviços que não são compatíveis com o standard IEC 61850 (figura 4.13).

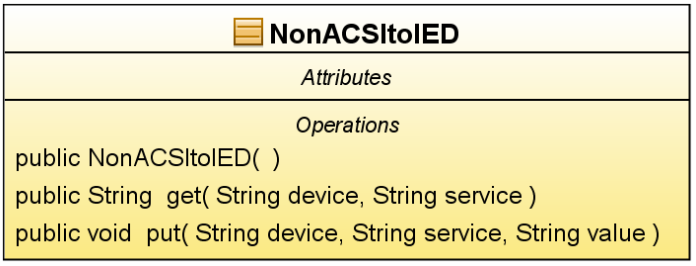

FIGURA 4.13: Diagrama de classe do componente *NonACSI.to.IED*.

4.5.2.2 *ACSI.Wrapper*

O componente *ACSI.Wrapper*, oferece ao componente *NEMO.IS.to.ACSI* funções para executar os serviços requiridos pela *Services Layer*, sendo estas: *put* e *get*, ou seja, para actuar ou para realizar operações no IED (figura 4.14). A classe *WrapperStruct* define

uma estrutura de dados através do qual é realizado o mapeamento interno, para os serviços que não são compatíveis com o modelo de dados ACSI.

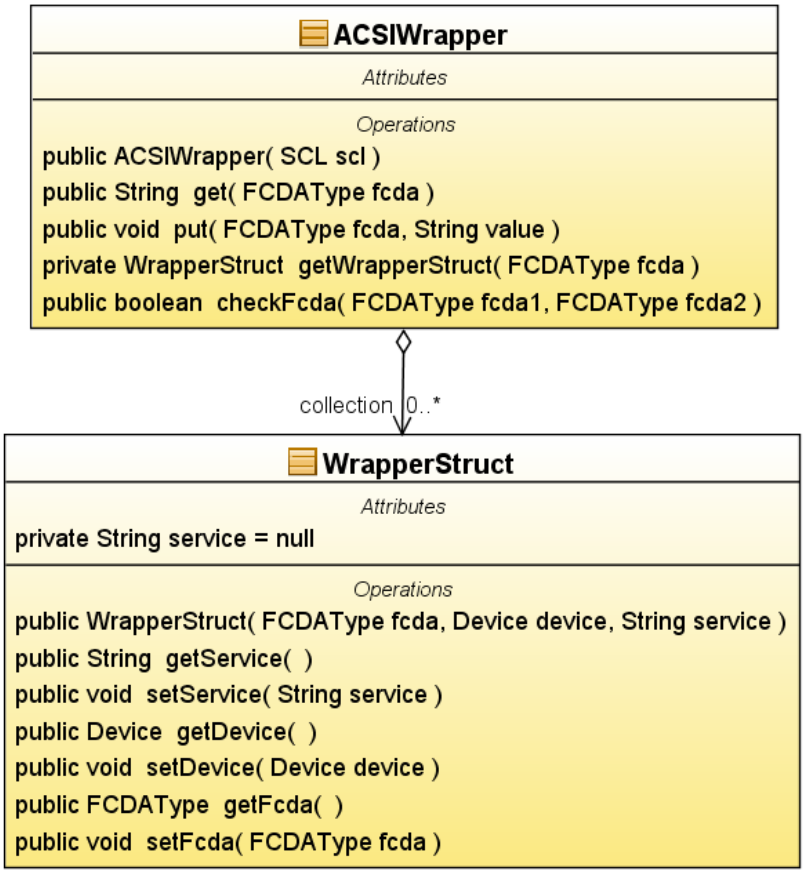

FIGURA 4.14: Diagrama de classe do componente *ACSI.Wrapper*.

4.5.2.3 *Communicator*

Este é o componente mais importante da *Communication Layer*, uma vez que é o responsável pela comunicação o IED ao qual o NEMO-C se encontra acoplado. A figura 4.15 apresenta os serviços que este componente oferece aos componentes *NonACSI.to.IED* e *ACSI.Wrapper* de modo a que estes possam executar as operações pedidas pela *Services Layer*. Nesta classe retirada do NEMO-C que se encontra conectado à pilha de combustível, pode-se visualizar as operações permitidas nos IEDs, sendo o *HydroBoy* o inversor da pilha de combustível e *Ballard* a pilha de combustível em si.

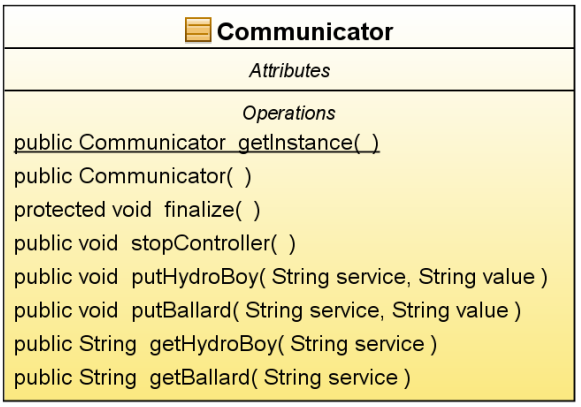

FIGURA 4.15: Diagrama de classe do componente *Communicator*.

4.5.3 Classes auxiliares

Esta subsecção apresenta as classes auxiliares das classes descritas anteriormente. Irão ser descritas as classes utilizadas para realizar a leitura dos ficheiros e as classes para a execução dos comandos directamente nos dispositivos físicos.

4.5.3.1 Classes para a leitura dos ficheiros

As classes *SAXParserNemoFile* (figura 4.16) e *SAXParserSCL* (figura 4.17) são utilizadas para realizar as leitura do NEMO-F e do ficheiro SCL, respectivamente. Com estas classes é possível realizar a leitura dos ficheiros para uma estrutura de dados, de modo a poder ser realizado a confirmação dos serviços invocados.

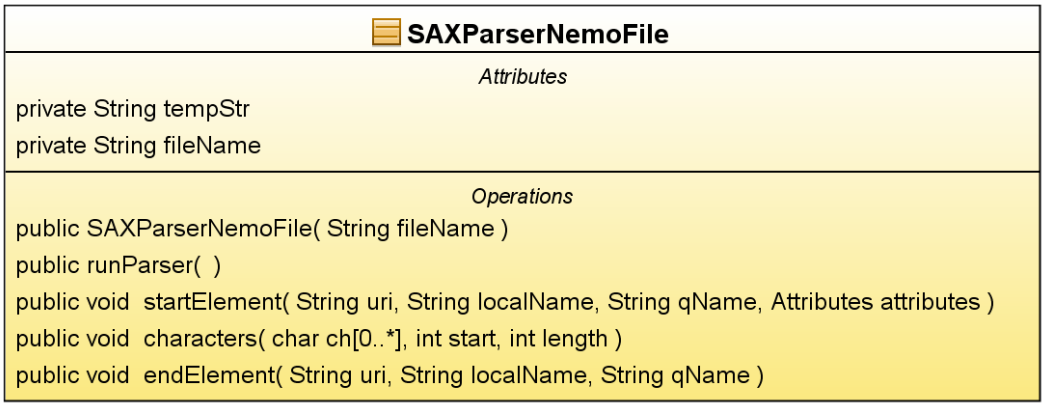

FIGURA 4.16: Diagrama de classe *SAXParserNemoFile*.

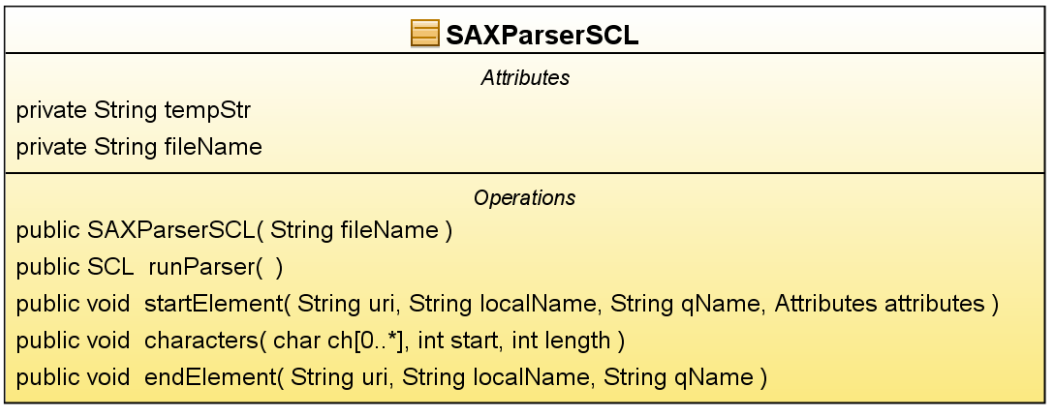

FIGURA 4.17: Diagrama de classe *SAXParserSCL*.

4.5.3.2 Classes para a execução de comandos

As classes *MasterHydroBoy* (figura 4.18) e *ReadBallard* (figura 4.16) são utilizados pelo componente *Communicator* de modo a aceder aos respectivos IEDs. A principal diferença entre as duas classes deve-se ao facto de os valores da pilha serem lidos periodicamente e quando a função de leitura é invocada os valores lidos anteriormente são devolvidos, enquanto os valores do inversor são lidos assincronamente, ou seja, quando a função de leitura é invocada pela *Services Layer*, o *Communicator* vai chamar a função de leitura do inversor.

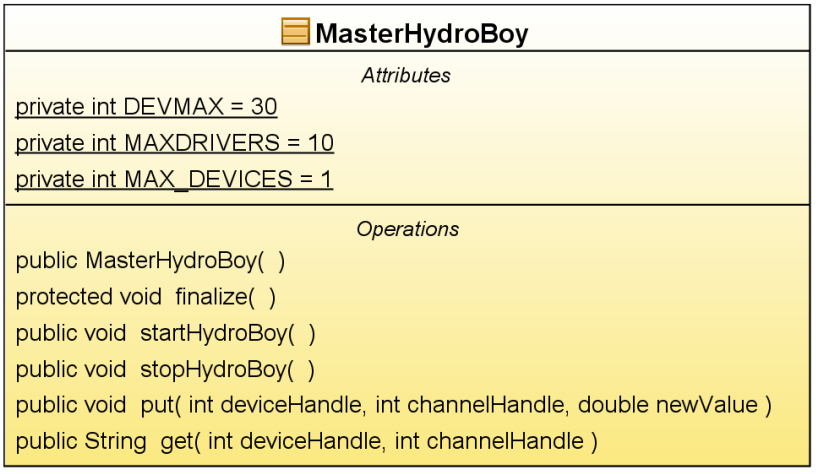

FIGURA 4.18: Diagrama de classe *MasterHydroBoy*.

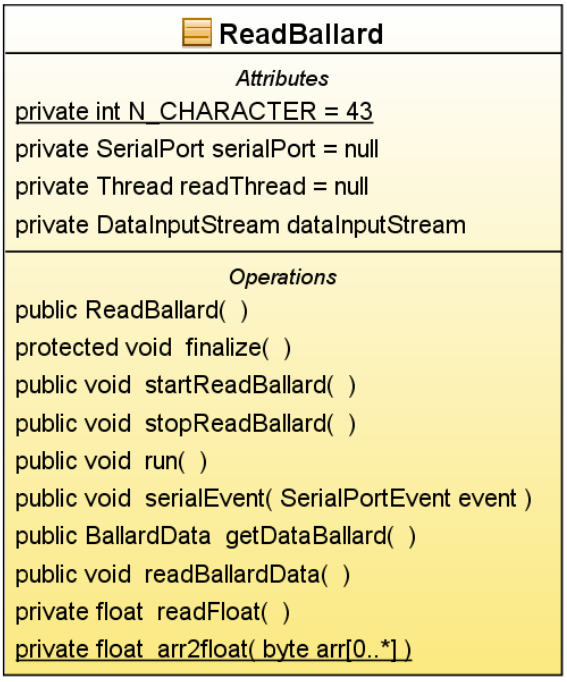

FIGURA 4.19: Diagrama de classe *ReadBallard*.

4.6 Visão Dinâmica

Em contraste com a visão estática, a visão dinâmica apresenta as interacções entre os objectos e no interior dos objectos, apresentado a resposta de um sistema às acções dos utilizadores. A visão dinâmica dos objectos é expressa a partir dos diagramas de sequência da linguagem UML. Cada um dos diagramas de sequência representa uma funcionalidade dos diagramas de caso de uso.

Nas subsecções seguintes irão ser mostradas as actividades entre os objectos e no interior dos mesmos.

4.6.1 Visão Dinâmica entre Objectos

Nas subsecções seguintes irão ser mostrados dois diagramas de casos de uso ambos referentes ao NEMO-C, com as interacções **Disponibilizar ficheiros** e **Realizar leitura de valores**.

4.6.1.1 Diagrama de sequência Disponibilizar ficheiros

O diagrama de sequência **Disponibilizar ficheiros** mostra a interacção entre o NEMO-V e o NEMO-C, quando o NEMO-V pede os ficheiros ao NEMO-C (figura 4.20). O Utilizador interage com o NEMO-V de modo a solicitar os ficheiros ao NEMO-C, o qual responde com os respectivos ficheiros, ou com uma mensagem de erro, caso não tenha conseguido executar o pedido.

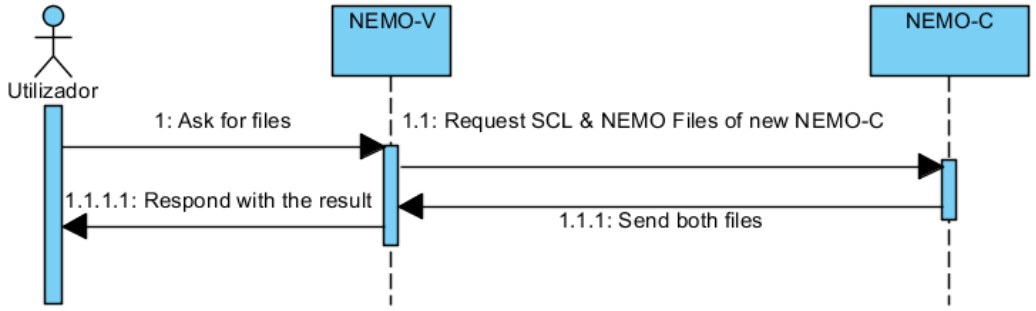

FIGURA 4.20: Diagrama de sequência **Disponibilizar ficheiros** do NEMO-C.

4.6.1.2 Diagrama de sequência Realizar leitura de valores

O diagrama de sequência **Realizar leitura de valores** mostra a interacção entre o NEMO-V e o NEMO-C, quando o NEMO-V precisa de actuar num IED que está a ser virtualizado por um determinado NEMO-C (figura 4.21). O Utilizador interage com o NEMO-V de modo a solicitar a execução de um determinado NEMO IS. Insere os dados necessários para a invocação do serviço, o qual será invocado no NEMO-C que irá, através do seu mapeamento interno, actuar no IED. Caso a operação seja bem sucedida, o NEMO-C responde com o resultado da operação, ou com uma mensagem de erro caso não tenha conseguido executar o serviço.

4.6.2 Visão Dinâmica Intra-Objectos

Nas subsecções seguintes irão ser mostrados dois diagramas que mostram a visão dinâmica intra-objectos referentes ao NEMO-C e ao NEMO-V. Os diagramas descrevem o comportamento intra-objecto através de diagramas de actividade UML.

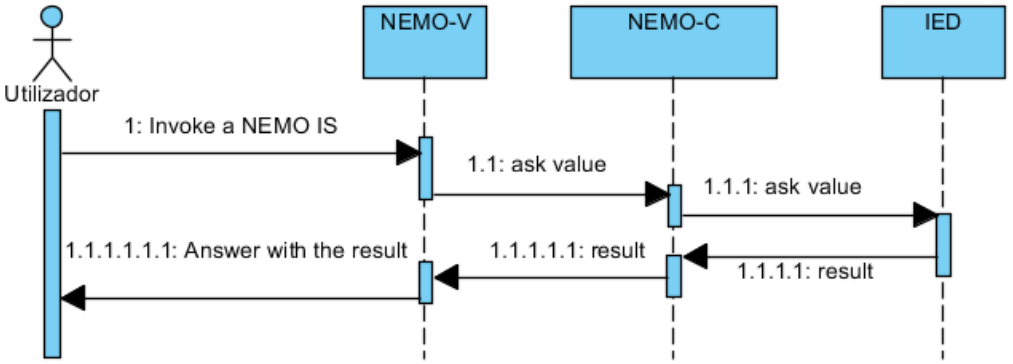

FIGURA 4.21: Diagrama de sequência **Realizar leitura de valores** do NEMO-C.

4.6.2.1 Diagrama de Actividade do NEMO-C

O comportamento interno do NEMO-C está descrito na figura 4.22. Após a leitura do ficheiro SCL e do NEMO-F, o NEMO-C encontra-se disponível para receber pedidos. Esses pedidos podem ser tanto de serviços ACSI como de serviços não ACSI. O pedido de execução de um determinado serviço NEMO, segue dois caminhos possíveis dependendo do nível de compatibilidade com o ACSI do serviço invocado.

Se o serviço for ACSI, irá ser despoletar o componente *NEMO.IS.to.ACSI*, o qual irá encaminhar de acordo com IED ao qual o NEMO-C se encontra conectado, para o *ACSI.Wrapper* se o IED não for compatível com o standard, ou irá invocar o serviço ACSI directamente no IED através do protocolo ACSI, se o IED for compatível com o standard IEC 61850. O *ACSI.Wrapper* irá realizar o mapeamento interno necessário para responder ao *NEMO IS* pretendido. Após o mapeamento, o serviço será invocado no IED e através do protocolo de comunicação necessário.

Por outro lado, se for um serviço não ACSI irá ser encaminhado para o componente *NEMO.IS.to.NonACSI*, o qual após verificar a sua exequibilidade, irá encaminhar o *NEMO IS* para o componente *NonACSI.to.IED*. Este componente irá executar o serviço através do seu mapeamento interno de modo a responder ao serviço requerido, após ter comunicado com o IED através do seu protocolo proprietário.

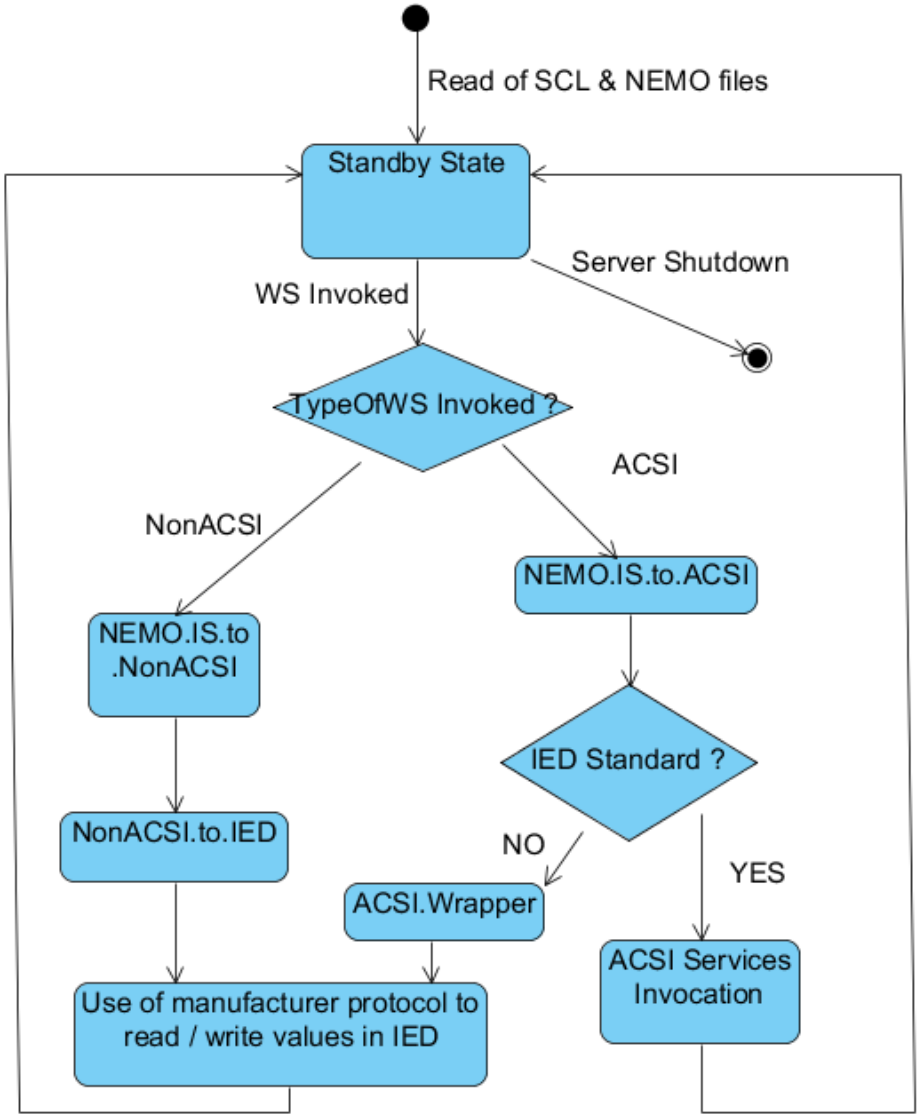

FIGURA 4.22: Diagrama de actividade do NEMO-C.

4.6.2.2 Diagrama de Actividade do NEMO-V

O comportamento interno do NEMO-V está descrito na figura 4.23. Após o início do NEMO-V, fica num estado *Standby State* em que pode receber qualquer pedido de operação: desligar o NEMO-V (*Stop NEMO-V*); descobrir IEDs (*Discovery*); invocar um serviço compatível com o standard (*Fill FCDA*) e invocar um serviço não compatível com o standard (*Fill NEMO IS*).

- *Stop NEMO-V:* Desligar o NEMO-V. Ao desligar o NEMO-V, já não será possível ao utilizador visualizar quando se conectam novos dispositivos.

- *Discovery:* Descobrir quais os IEDs relacionados com energia que se encontram conectados à NEMO SN. Após a descoberta de novos IEDs, ou se algum IED se tiver desconectado, a interface do NEMO-V é actualizada, permitindo a visualização em tempo real de todos os IEDs que é possível monitorizar e/ou controlar.

- *Fill FCDA:* Invocar um serviço compatível com o standard IEC 61850. Ao invocar este serviço é necessário o utilizador escolher quais os parâmetros do FCDA para depois proceder a invocação do serviço. Depois, a interface é actualizada com o resultado do serviço, ou com uma mensagem de erro, caso não tenha sido possível executar o serviço.

- *Fill NEMO IS:* Invocar um serviço não compatível com o standard IEC 61850. Para a invocação deste serviço é necessário o utilizador escolher qual o NEMO IS que quer invocar. Depois da invocação do serviço, a interface é actualizada com o resultado do serviço, ou com uma mensagem de erro, caso não tenha sido possível executar o serviço.

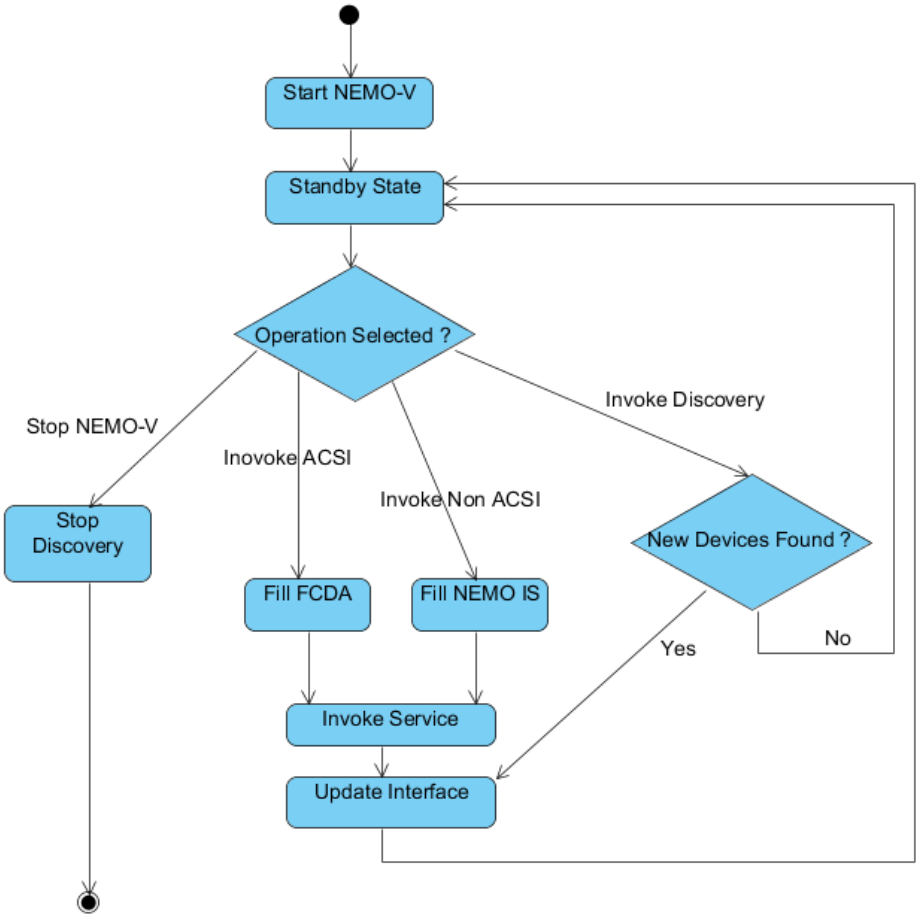

FIGURA 4.23: Diagrama de actividade do NEMO-C.

4.7 Exemplo de Validação do Sistema

Esta secção apresenta um exemplo de execução da infra-estrutura implementada. Não cobre todas as funcionalidades descritas nesta dissertação, mas oferece uma perspectiva geral sobre o que foi implementado nesta instância NEMO. A interface do NEMO-V permite ver todos os IEDs que estão conectados na NEMO SN (figura 4.24).

Vamos considerar que o utilizador quer solicitar um serviço não ACSI disponível para o IED que quer monitorizar e/ou controlar. Para o NEMO-V saber que serviços permite ao utilizador invocar, necessita do ficheiro SCL e do NEMO-F do IED. O utilizador terá que realizar a transferência dos ficheiros através da interface da figura 4.25.

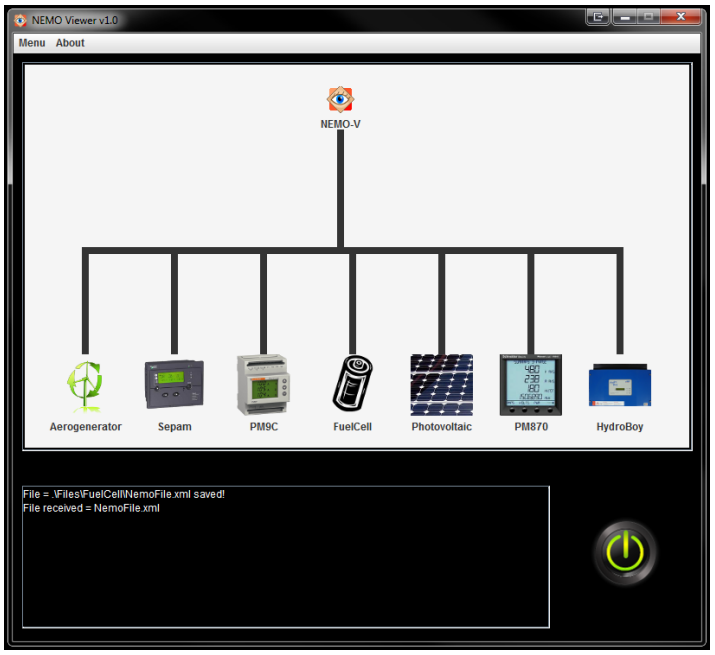

FIGURA 4.24: Interface do NEMO-V.

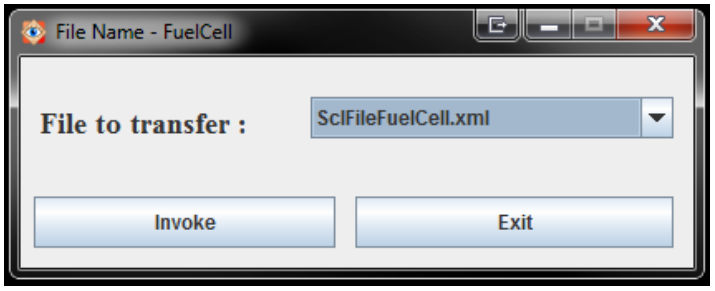

FIGURA 4.25: Interface de transferência de ficheiros do NEMO-V.

Esta interface permite que o utilizador realize a transferência dos ficheiros necessários para o funcionamento do NEMO-V. Após a transferência dos ficheiros, o utilizador pode então seleccionar a invocação de um serviço não ACSI do IED. A interface apresentada será a da figura 4.26 que disponibiliza ao utilizador os serviços da tabela 4.2.

Considere-se que o utilizador quer realizar uma operação de monitorização, por exemplo, da pressão de combustível da pilha de combustível. Esta operação inicia-se com a invocação do serviço *GetNonIEC()* tendo como parâmetro o identificador de serviço NEMO *HydrogenPressure*.

Este pedido vai ser recebido pelo componente *NEMO.IS* e uma vez que é identificado

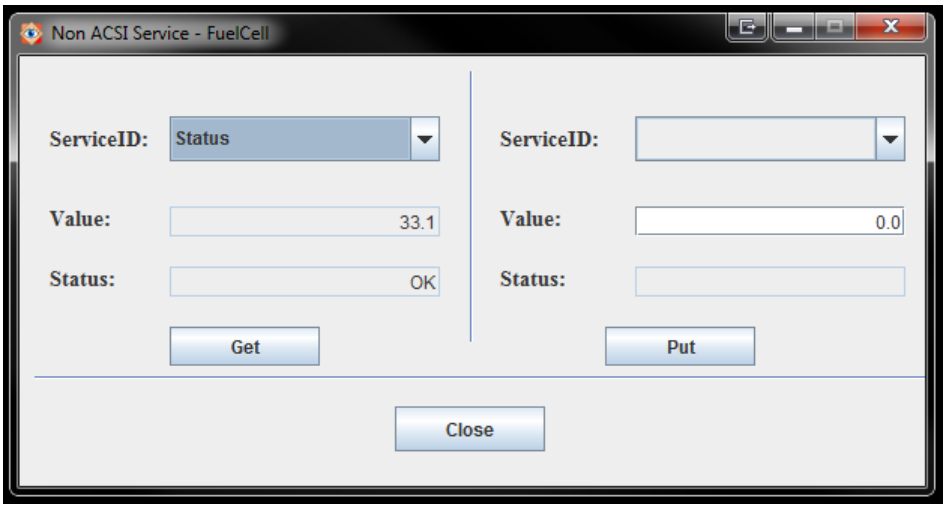

FIGURA 4.26: Interface do NEMO-V para serviços não ACSI.

TABELA 4.2: Serviços não ACSI do inversor (*HydroBoy*) e da pilha de combustível (*FuelCell*).

Device	Type	NEMO Service Identifier	Unit
HydroBoy	Get	SerialNumber	n/A
	Get	DCVtgStr	V
	Get	TStop	s
	GetPut	DCCurChgSp	mA/s
	GetPut	EnOp	n/A
	GetPut	DCCurStPt	mA
	Get	Plimit	W
	Get	FrqDifMax	Hz/s
	Get	HwVer	n/A
	Get	Firmware-BFR	n/A
	Get	Firmware-SRR	n/A
	Get	Mode	n/A
	Get	Storage	n/A
NEXA Fuel Cell	Get	Status	n/A
	Get	FailCode	n/A
	Get	WarningBitmap	n/A
	Get	HydrogenPressure	barg
	Get	HydrogenConcentration	%
	Get	ComulativeHydrogenConsumption	L
	Get	OxygenConcentration	L

como um serviço não compatível com o standard, ele é encaminhado para o componente *NEMO.IS.to.NonACSI* onde a sua viabilidade é verificada. Se o NEMO-C reconhece o serviço solicitado, será enviado ao componente *NonACSI.to.IED*, que irá realizar a operação solicitada através do componente *Communicator*, de acordo com o mapeamento interno do NEMO-C. Caso contrário, o serviço será negado. O resultado da invocação

do serviço será então apresentado na interface do NEMO-V. Após uma execução de serviço correcta, o seu resultado será disponibilizado ao utilizador como mostra a figura 4.26. Por outro lado, caso tenha existido algum problema na execução, o campo *Status* da interface irá aparecer com a mensagem de erro *ERROR*, em vez de *OK*.

A interface para a invocação de um serviço não ACSI permite tanto operações de monitorização como de controlo. Basta o utilizador indicar qual o valor para escrever no IED e seleccionar a operação *Put*, o que irá despoletar o serviço *PutNonIEC()*. A execução interna do serviço é igual à descrita anteriormente, mas em vez de uma operação de leitura no IED será realizada uma operação de escrita.

Vamos agora considerar a monitorização da temperatura da pilha de combustível, realizado pelo serviço compatível com o standard ACSI, *GetIEC(STMP1.Tmp.mag[MX])*, como mostra a figura 4.27.

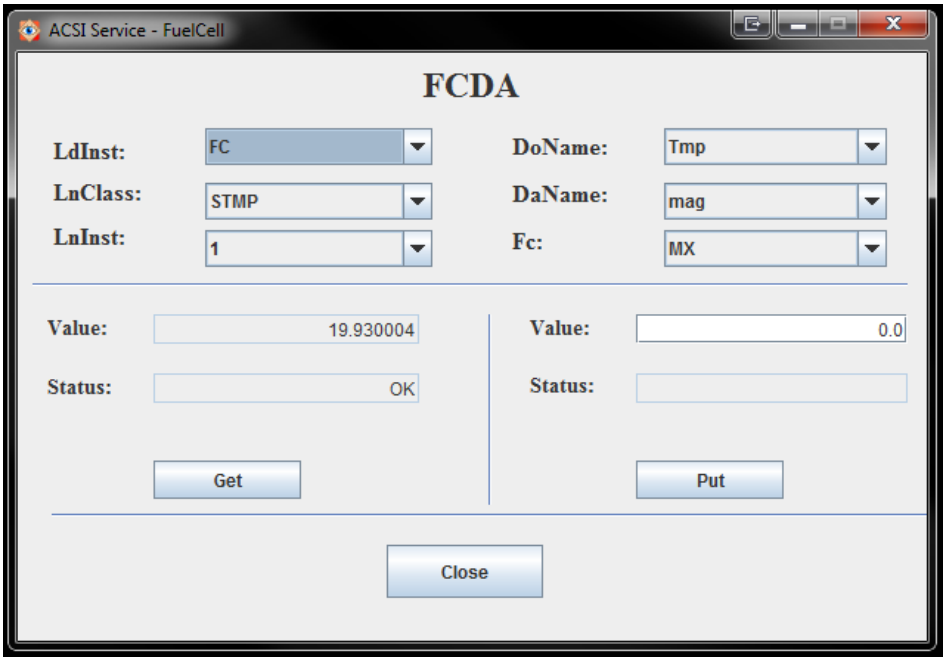

FIGURA 4.27: Interface do NEMO-V para serviços ACSI.

Como descrito anteriormente, o NEMO-V realiza um pedido do serviço para o *NEMO.IS* mas desta vez o pedido é encaminhado para o componente *NEMO.IS.to.ACSI*, que verifica se o parâmetro FCDA do serviço solicitado está em conformidade com o ficheiro SCL do IED. Se o IED de destino que está a ser virtualizado pelo NEMO-C é um dispositivo compatível com o standard, a operação é realizada de acordo com os protocolos IEC 61850. Caso contrário, o componente *ACSI.Wrapper* será despoletado e o serviço

é executado de acordo com o mapeamento interno FCDA do NEMO-C. Através do mapeamento interno, um serviço específico do *Comunicator* é invocado para se obter a operação de monitorização pretendida. Neste caso, como se pode observar na figura 4.27, a temperatura da pilha de combustível é de 19,93 žC.

Como se pode observar no decorrer deste capítulo é possível aceder aos IEDs por intermédio do NEMO-C. Através do ficheiro SCL de cada IED, o NEMO-V vai permitir ao utilizador preencher o FCDA de modo a poder realizar operações. Para os serviços que não são compatíveis com o standard IEC 61850, o NEMO-V vai ler os NEMO-F do NEMO-C, sabendo assim quais são os NEMO IS disponíveis para aquele IED de modo a permitir ao utilizador escolher o serviço pretendido. O reconhecimento de quando um NEMO-C se conecta ou desconecta da NEMO SN também é possível devido à procura periódica.

Capítulo 5

Conclusões e Perspectivas Futuras

Este capítulo apresenta uma síntese do trabalho efectuado, atendendo aos objectivos anunciados no primeiro capítulo. Posteriormente, comentam-se os contributos deste projecto e possíveis trabalhos futuros.

5.1 Síntese Geral

O homem começou a tomar consciência que a sua dependência de energia estava a atingir o limite, levando-o assim a intensificar a procura de fontes de energia renovável, para diminuir o impacto do elevado consumo no meio ambiente, quer por emissões de gases de efeito de estufa, quer pela extracção dos recursos fósseis.

O trabalho apresentado nesta dissertação, tem como objectivo o desenvolvimento de uma infra-estrutura de software, que sirva de apoio à produção, consumo e distribuição de energia. A infra-estrutura tem como objectivo, permitir a monitorização e controlo de diversos dispositivos, existentes nas diferentes áreas relacionadas com a energia. De modo a cumprir esse objectivo, a infra-estrutura implementada apresenta as seguintes funcionalidades:

- Permite a descoberta de todos os dispositivos existentes na rede de energia que estejam virtualizados pelo NEMO-C.

- Permite que a comunicação com os dispositivos seja feita através de um standard, independentemente do dispositivo com que se quer interagir.
- Permite realizar a monitorização e controlo dos dispositivos.
- Permite criar uma rede de software para controlar uma rede de dispositivos relacionados com a energia.

5.2 Contribuição da Investigação

Na presente dissertação foram mostradas diversas situações em que a gestão de dispositivos relacionados com a energia, são deveras importantes, quer na parte da poupança da energia, quer na optimização do consumo da mesma e as ICTs são ferramentas que estão a ser exploradas nesse sentido.

O software desenvolvido nesta dissertação apresenta-se como uma contribuição no suporte a um sistema de gestão de energia que necessite de controlar dispositivos nas diversas áreas relacionadas com a energia. A implementação da infra-estrutura de software mostra que é possível gerir dispositivos de baixo nível através do paradigma SOA, de modo a que os dispositivos disponibilizem serviços, os quais podem ser acedidos pelo sistema gestor de energia.

Através do uso dos *standards* DPWS e IEC 61850 foi possível desenvolver uma infra-estrutura de software distribuída que permite interoperabilidade dos dispositivos relacionados com energia. A infra-estrutura desenvolvida abrange todo o processo energético desde a geração até ao consumo.

Como já tinha sido referido na secção 1.4, esta dissertação foi desenvolvido no contexto do projecto NEMO que ainda se encontra em curso, permitindo assim que algumas características que não foram implementadas e que estão descritas na secção seguinte, possam ainda vir a ser concretizadas.

Os resultados desta dissertação foram aceites para publicação nas conferências:

- ISIE 2011: *"A standard-based software infrastructure to support energy efficiency using renewable energy sources"* ([27]).

- INDIN 2011: *"DPWS as Specific Communication Service Mapping for IEC 61850"* ([29]).

5.3 Trabalhos Futuros

A infra-estrutura de software apresenta diversas características que não chegaram a ser testadas e conceitos que não chegaram a ser implementados, nomeadamente:

- A conexão de um NEMO-C a um dispositivo que fosse possível comunicar directamente com o standard IEC 61850, não chegou a ser testada.
- A conexão do NEMO-C a outros dispositivos relacionados com a energia (por exemplo, os inversores dos painéis fotovoltaicos e do gerador eólico), de modo a poder ter um software mais versátil e universal.
- O NEMO-K ser capaz de realizar o reconhecimento automático e aprender através de redes neuronais, ou de algo semelhante, o tipo de dispositivo ao NEMO-C se encontra conectado.
- O desenvolvimento de um NEMO-V como um aplicativo para um telemóvel.

Bibliografia

[1] Baigent, D., Adamiak, M., Mackiewicz, R., and SISCO, G. (2005). Iec 61850 communication networks and systems in substations: An overview for users. In *Proceedings of the VIII Simposio "Iberoamericano Sobre Proteccion de Sistemas Electricos de Potencia", Monterey, Mexico*. Citeseer.

[2] Bailey, D. and Wright, E. (2003). *Practical SCADA for industry*. Newnes, Elsevier, Linacre House, Jordan Hill, Oxford OX2 8DP.

[3] Barry, D. K. (2010). Service-oriented architecture (soa) definition. `http://www.service-architecture.com/web-services/articles/service-oriented_architecture_soa_definition.html`.

[4] Bell, M. (2008). *Service-Oriented Modeling (SOA): Service Analysis, Design, and Architecture*. Wiley.

[5] Bloomberg, J. and Schmelzer, R. (2006). *Service orient or be doomed!: how service orientation will change your business*. John Wiley & Sons Inc.

[6] Booth, D., Haas, H., McCabe, F., Newcomer, E., Champion, M., Ferris, C., and Orchard, D. (2004). Web services architecture. `http://www.w3.org/TR/ws-arch/`.

[7] Bray, T., Paoli, J., Sperberg-McQueen, C. M., Maler, E., and Yergeau, F. (2006). Extensible markup language (xml) 1.0 (fourth edition). `http://www.w3.org/TR/2006/REC-xml-20060816`.

[8] Cândido, G., Jammes, F., Barata, J., and Colombo, A. (2009). Generic management services for dpws-enabled devices. In *Industrial Electronics, 2009. IECON '09. 35th Annual Conference of IEEE*, pages 3931 –3936.

[9] Chakraborty, S., Weiss, M., and Simões, M. (2007). Distributed intelligent energy management system for a single-phase high-frequency ac microgrid. *Industrial Electronics, IEEE Transactions on*, 54(1):97–109.

[10] Choi, C.-S., Park, W.-K., Han, J.-S., and Lee, I.-W. (2010). The architecture and implementation of proactive green home energy management system. In *Information and Communication Technology Convergence (ICTC), 2010 International Conference on*, pages 457 –458.

[11] Christensen, E., Curbera, F., Meredith, G., and Weerawarana, S. (2001). Web services description language (wsdl) 1.1. `http://www.w3.org/TR/2001/NOTE-wsdl-20010315`.

[12] Daneels, A. and Salter, W. (1999). What is scada? In *Proceedings on the International Conference on Accelerator and Large Experimental Physics Control System, Trieste, Italy*.

[13] Driscoll, D. and Mensch, A. (2009). Devices profile for web services version 1.1 specification. `http://docs.oasis-open.org/ws-dd/ns/dpws/2009/01`.

[14] EMS (2011). Energy management systems. `http://www.bsigroup.co.uk/en/Assessment-and-Certification-services/Management-systems/Standards-and-Schemes/EN-16001-Energy-Management-Systems/`.

[15] EnergyCAP (2011). Energycap ®enterprise. `http://www.energycap.com/products/energycap-enterprise`.

[16] Enterprise energy management (2011). Enterprise energy management is hard. we make it easy. `http://ecomponentstech.com/`.

[17] Erl, T. (2005). *Service-oriented architecture: concepts, technology, and design*. Prentice Hall PTR.

[18] Gasser, U. and Palfrey, J. (2007). *Breaking Down Digital Barriers: When and How ICT Interoperability Drives Innovation*. Berkman Center for Internet & Society, Harvard Law School.

[19] Gosling, J. and McGilton, H. (2011). The java language environment. `http://net.uom.gr/Books/Manuals/langenviron-a4.pdf`.

[20] Industrial Revolution (2011). Industrial revolution. `http://www.britannica.com/EBchecked/topic/287086/Industrial-Revolution`.

[21] Interoperability (2011). What is interoperability? `http://searchsoa.techtarget.com/definition/interoperability`.

[22] ITEA SIRENA project (2011). Welcome to the itea sirena project. `http://www.sirena-itea.org/`.

[23] ITEA SODA project (2011). Welcome to the itea soda project. `http://www.soda-itea.org/`.

[24] Jammes, F., Smit, H., Lastra, J. L. M., and Delamer, I. (2005). Orchestration of service-oriented manufacturing processes. In *Emerging Technologies and Factory Automation, 2005. ETFA 2005. 10th IEEE Conference on*, volume 1, pages 8 pp. –624.

[25] Jamod (2010). Java modbus library (jamod). `http://jamod.sourceforge.net/`.

[26] Khayyam, H., Kouzani, A., and Hu, E. (2009). Reducing energy consumption of vehicle air conditioning system by an energy management system. In *Intelligent Vehicles Symposium, 2009 IEEE*, pages 752–757. IEEE.

[27] Lima, C., Gomes, V., Lima, J., Martins, J. a. F., Barata, J., Ribeiro, L., and Cândido, G. (2011a). A standard-based software infrastructure to support energy efficiency using renewable energy sources. *20th IEEE International Symposium on Industrial Electronics*.

[28] Lima, C., Martins, J. a. F., Barata, J., Ribeiro, L., and Cândido, G. (2010). Towards a service based infrastructure to improve efficiency into energy systems: the nemo&coded quest. *10th IFAC Workshop on Intelligent Manufacturing Systems*.

[29] Lima, J., Lima, C., Gomes, V., Martins, J. a. F., Barata, J., Ribeiro, L., and Cândido, G. (2011b). Dpws as specific communication service mapping for iec 61850. *IEEE 9th International Conference on Industrial Informatics*.

[30] Modbus (2011). The modbus organization. `http://www.modbus.org/`.

[31] Moreno, J., Ortuzar, M., and Dixon, J. (2006). Energy-management system for a hybrid electric vehicle, using ultracapacitors and neural networks. *Industrial Electronics, IEEE Transactions on*, 53(2):614 – 623.

[32] NetBeans (2010). Netbeans ide 6.9.1 release information. `http://netbeans.org/community/releases/69/index.html`.

[33] Ribeiro, L., Barata, J., and Mendes, P. (2008). Mas and soa: Complementary automation paradigms. In Azevedo, A., editor, *Innovation in Manufacturing Networks*, volume 266 of *IFIP International Federation for Information Processing*, pages 259–268. Springer Boston.

[34] Rosen, M., Lublinsky, B., Smith, K. T., and Balcer, M. J. (2008). *Applied SOA: Service-Oriented Architecture and Design Strategies*. Wiley.

[35] RXTX (2010). Rxtx: The prescription for transmission. `http://users.frii.com/jarvi/rxtx/index.html`.

[36] SCADA (2011). Scada systems. `http://www.scadasystems.net/`.

[37] SOA for Devices (2010). Service-oriented architecture for devices. `https://forge.soa4d.org/`.

[38] SOCRADES (2011). Socrades : Service-oriented cross-layer infrastructure for distributed smart embedded systems. `http://cordis.europa.eu/fetch?CALLER=PROJ_ICT&ACTION=D&CAT=PROJ&RCN=79520`.

[39] Standards Microsoft (2010). Standards in the ict industry. `http://download.microsoft.com/download/7/F/9/7F9FEC37-84FF-4020-A243-9BEBE70DC0B9/Standards_in_the_ICT_Industry.pdf`.

[40] Stum, K., Mosier, R., and Haasl, T. (1997). *Energy Management Systems-a Practical Guide*. Portland Energy Conservation Inc. (PECI), 921 SW Washington, Suite 312.

[41] Tac Vista (2011). System overview - tac vista. `http://www.global-download.schneider-electric.com/852577A4005D7372/all/A218853A9D9B7E56852578070068622E/$File/tac_vista_system_overview_brochure.pdf`.

[42] Tanenbaum, A. (2003). *Computer Networks*. Prentice Hall PTR, fourth edition edition.

[43] Temsco (2011). Total control energy and security management. `http://www.temsco.com/energy.htm`.

[44] Visual Paradigm (2011). Visual paradigm for uml. `http://www.visual-paradigm.com/product/vpuml/`.

[45] Zittrain, J. (2009). *The Future of the Internet–and how to Stop it*. Yale University Press, New Haven & London.

Apêndice A

DER da base de dados IEDs

O DER da base de dados IEDs não é possivel ser visualizado em apenas uma página, como tal, vai ser disponibilizado nas quatro figuras seguintes.

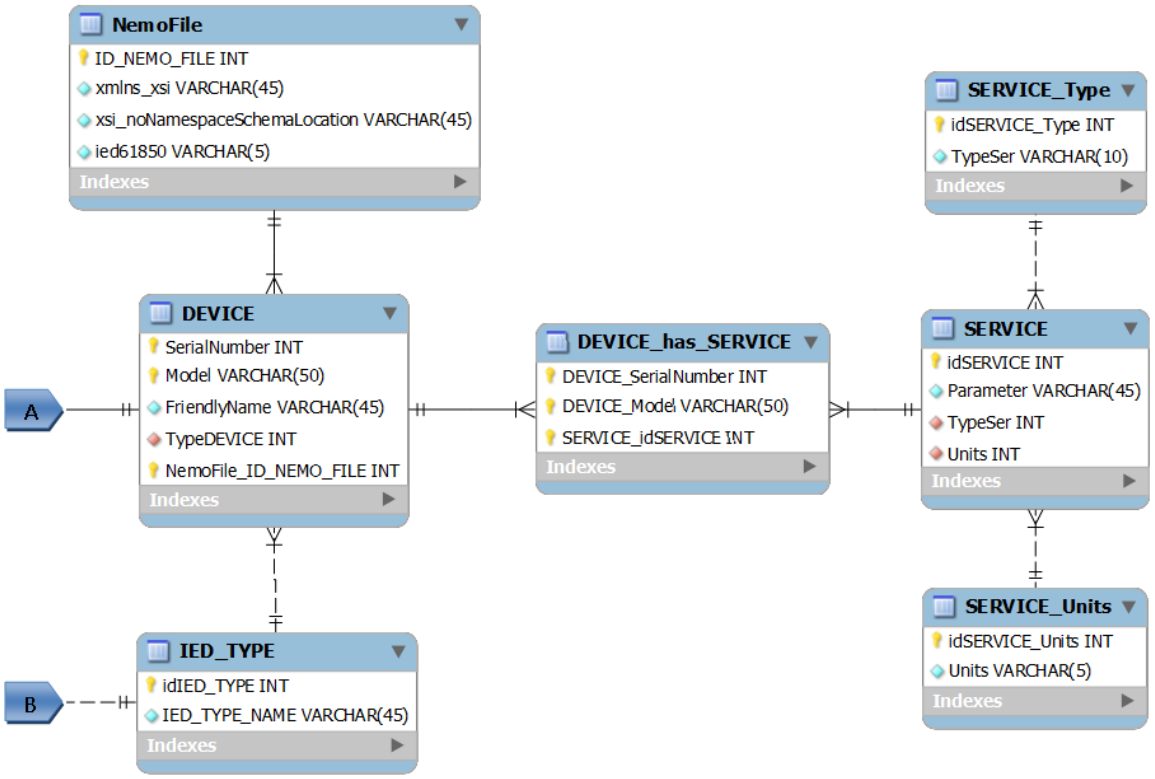

FIGURA A.1: DER da base de dados IEDs - Componente NEMO-F

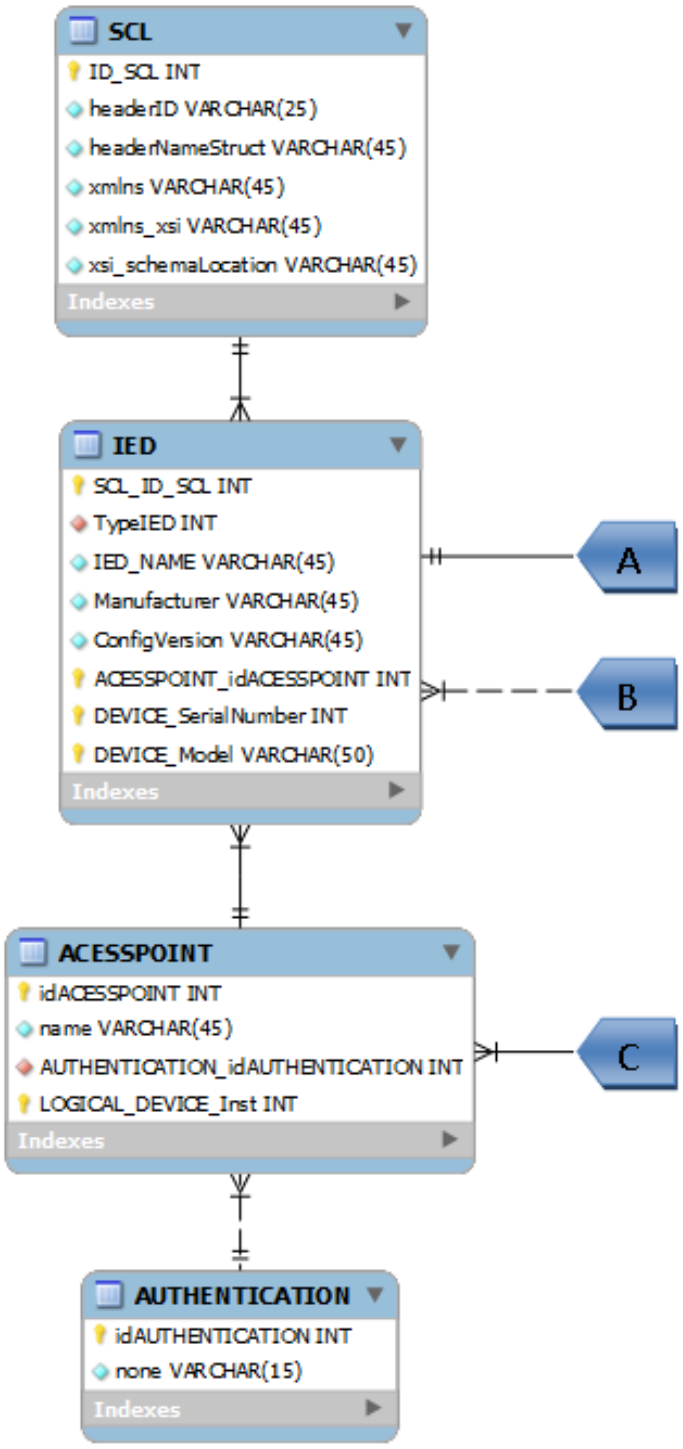

FIGURA A.2: DER da base de dados IEDs - Componente SCL

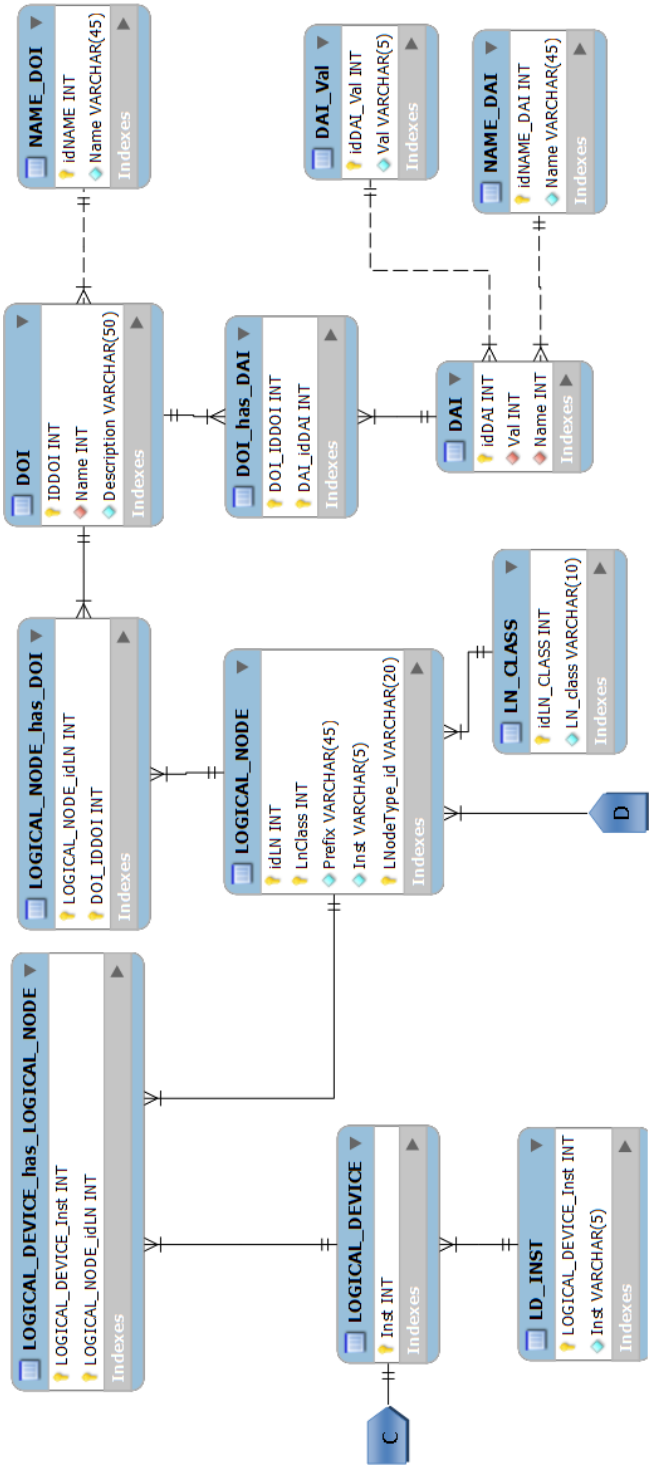

FIGURA A.3: DER da base de dados IEDs - Componente *SCL Heading*

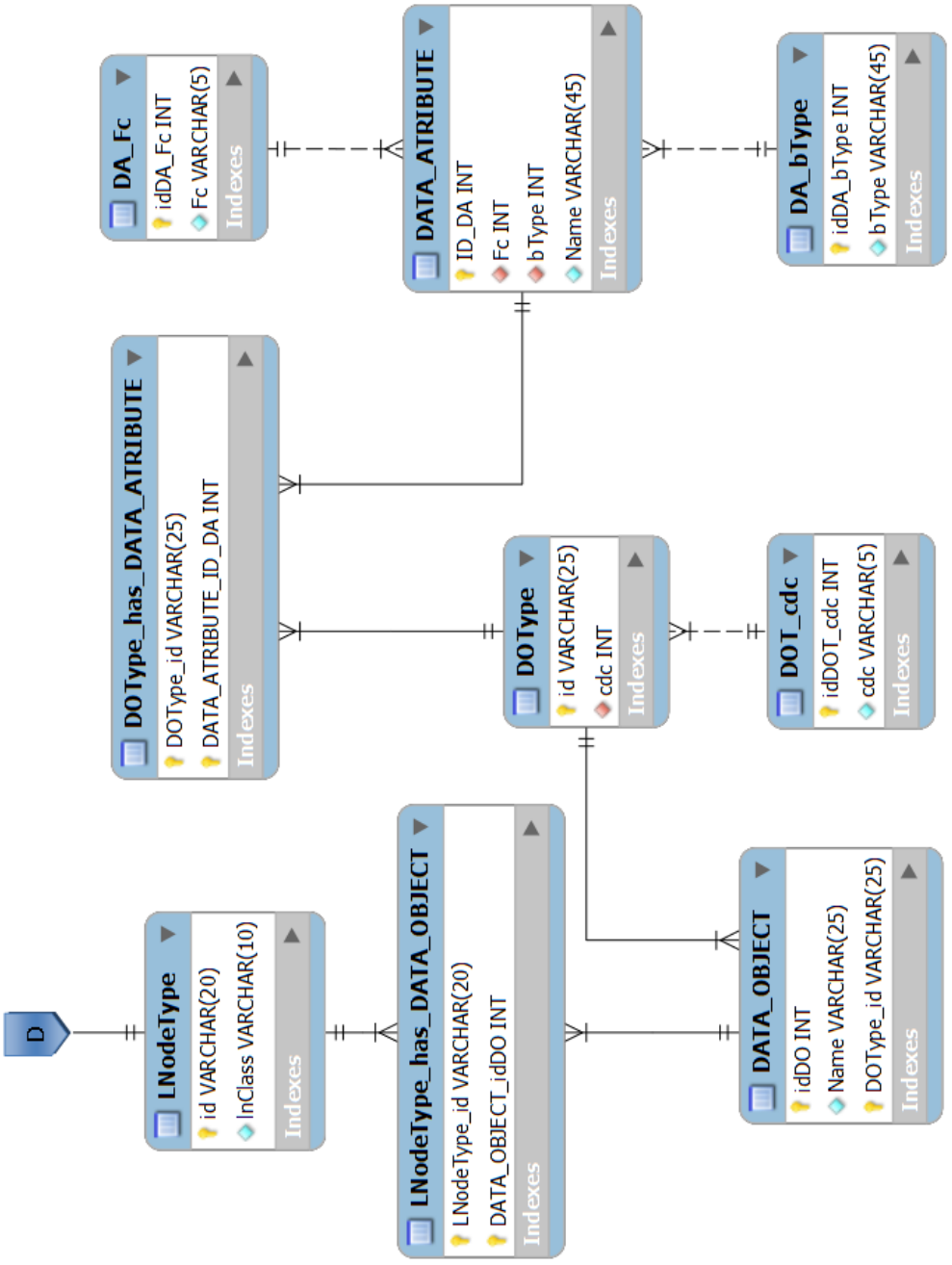

FIGURA A.4: DER da base de dados IEDs - Componente *Data Type Templates*

Apêndice B

WSDL do serviço DPWS *File Transfer*

```
<?xml version="1.0" encoding="UTF-8"?>
<wsdl:definitions name="FileTransfer"

    targetNamespace="http://www.nemo.com/NemoIS"
    xmlns:tns="http://www.nemo.com/NemoIS"
    xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap12/"
    xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:wse="http://schemas.xmlsoap.org/ws/2004/08/eventing">
    <wsdl:documentation>
        A simple transfer service.
    </wsdl:documentation>


    <!--
            TYPES
    -->


    <wsdl:types>
        <xsd:schema
            targetNamespace="http://www.nemo.com/NemoIS"
            elementFormDefault="qualified"
            xmlns:tns="http://www.nemo.com/NemoIS">
```

```
            <xsd:element name="FileType" type="xsd:base64Binary"/>
            <xsd:element name="FileNameType" type="xsd:string"/>

        </xsd:schema>
    </wsdl:types>

    <!--
            MESSAGES
    -->

    <wsdl:message name="FileRequestReqMsg">
        <wsdl:part name="FileName" element="tns:FileNameType"/>
    </wsdl:message>
    <wsdl:message name="FileRequestRespMsg">
        <wsdl:part name="File" element="tns:FileType"/>
    </wsdl:message>

    <!--
            PORT TYPES
    -->
    <wsdl:portType name="Transfer" wse:EventSource="true">
        <wsdl:documentation>
                This port type defines the file transfer.
        </wsdl:documentation>
        <wsdl:operation name="TransferFile">
            <wsdl:documentation>
                File request service.
            </wsdl:documentation>
            <wsdl:input message="tns:FileRequestReqMsg"/>
            <wsdl:output message="tns:FileRequestRespMsg"/>
        </wsdl:operation>
    </wsdl:portType>

    <!--
            BINDINGS
    -->
    <wsdl:binding name="Transfer" type="tns:Transfer">
        <soap:binding style="document"
                        transport="http://schemas.xmlsoap.org/soap/http" />
        <wsdl:operation name="TransferFile">
```

```
            <soap:operation soapAction="http://www.nemo.com/NemoIS/TransferFile"
                        style="document"/>
            <wsdl:input>
                <soap:body use="literal" />
            </wsdl:input>
            <wsdl:output>
                <soap:body use="literal" />
            </wsdl:output>
        </wsdl:operation>
    </wsdl:binding>

    <!--
            Services (Not used by the toolkit)
    -->
    <wsdl:service name="FileTransfer">
        <wsdl:documentation>
            This is the Nemo Internal Services Web Service,
            defining the 'FileTransfer'.
        </wsdl:documentation>
        <wsdl:port name="Transfer" binding="tns:Transfer">
            <wsdl:documentation>
                Port that defines the transfer of a file.
            </wsdl:documentation>
        </wsdl:port>
    </wsdl:service>
</wsdl:definitions>
```