POLITÉCNICO DE LEIRIA | ESCOLA SUPERIOR DE TECNOLOGIA E GESTÃO

Tooling 4G - Advanced Tools for Smart Manufacturing

Mestrado em Engenharia Informática - Computação Móvel

Alberto Luís Bastos Trindade

Leiria, novembro de 2020

POLITÉCNICO DE LEIRIA | ESCOLA SUPERIOR DE TECNOLOGIA E GESTÃO

Tooling 4G - Advanced Tools for Smart Manufacturing

Mestrado em Engenharia Informática - Computação Móvel

Alberto Luís Bastos Trindade

Dissertação/Trabalho de Projeto realizada/o sob a orientação do Professor Doutor Ricardo Martinho e do Professor Doutor Rui Rijo.

Leiria, novembro 2020

Originalidade e Direitos de Autor

O presente relatório de projeto é original, elaborado unicamente para este fim, tendo sido devidamente citados todos os autores cujos estudos e publicações contribuíram para o elaborar.

Reproduções parciais deste documento serão autorizadas na condição de que seja mencionado o Autor e feita referência ao ciclo de estudos no âmbito do qual o mesmo foi realizado, a saber, Curso de Mestrado em Engenharia Informática – Computação Móvel, no ano letivo 2019/2020, da Escola Superior de Tecnologia e Gestão do Instituto Politécnico de Leiria, Portugal, e, bem assim, à data das provas públicas que visaram a avaliação destes trabalhos.

Agradecimentos

Uma tese de mestrado é um projeto bastante solitário e este em particular não poderia ter sido concluído sem o apoio direto ou indireto de muitas pessoas, nas boas e piores fases desta tese.

Gostaria de primeiramente agradecer aos meus orientadores, professor doutor Ricardo Martinho e professor doutor Rui Rijo, por toda a simpatia, toda a disponibilidade e também toda a transferência de *know how* que me permitiram produzir esta tese.

Gostaria de agradecer ao Engenheiro Ricardo Silva, ex-gestor de projeto da inCentea e ao Engenheiro Nelson Marques gestor de unidade da inCentea por toda a ajuda técnica prestada de forma a conseguir implementar o projeto que deu origem a esta tese.

Gostaria também de agradecer ao professor doutor Carlos Neves, por todos os esclarecimentos referentes ao projeto.

À minha família, pai e mãe, que sempre me apoiaram e mostraram interesse pelo meu trabalho. Foram eles que sempre me ouviram e apoiaram, nos momentos de desânimo e de alento.

Aos meus amigos, que sempre se mostraram interessados na tese e me apoiaram incondicionalmente.

Resumo

O setor industrial, sendo um dos grandes impulsionadores do desenvolvimento macroeconómico global, torna-se também um objeto interessante de investimento por parte de entidades públicas e privadas. Face à crescente procura por produtos e serviços que possam suprir as necessidades de toda a população mundial, torna-se necessário a criação em massa de produtos com o mínimo de gasto (*zero waste*) e o máximo de eficiência. Num chão de fábrica, este objetivo exige soluções que permitam a recolha e a troca de dados dos vários sistemas de informação que apoiam os seus processos de negócio. Estes dados permitem monitorizar as operações, melhorar o planeamento, melhorar os processos e reduzir os erros e defeitos de fabrico. O presente trabalho propõe uma arquitetura de software para a monitorização de um chão de fábrica, baseada no protocolo de comunicação máquina-máquina aberto *Open Platform Communications - Unified Architecture* (OPC-UA) desenvolvido pela OPC *Foundation*. Este protocolo posiciona-se para vir a ser um *standard de facto* da indústria. Os trabalhos iniciaram-se com um levantamento do estado da arte por forma a apurar como a digitalização das empresas fabris pode maximizar os indicadores de produtividade. Com base nos desafios identificados foi realizado um estudo de caso da digitalização de uma empresa de moldes com o desenvolvimento de uma *framework* que utiliza o protocolo industrial OPC-UA como forma de integração entre um sistema de informação do tipo *Enterprise Resource Planning* (ERP) e todos os serviços que compõem uma arquitetura de software industrial, permitindo observar em tempo real indicadores de desempenho, bem como digitalizar a gestão de ordens de fabrico num chão de fábrica. O presente projeto insere-se no projeto Mobilizador TOOLING4G – *Advanced Tools for Smart Manufacturing* (~ 7M€ e 32 parceiros) e enquadra-se no *cluster engineering & tooling*, que integra uma cadeia de valor alargada (do *design* ao produto final), para responder a clientes globais que cada vez mais pretendem soluções chave-na-mão.

Palavras-chave: Indústria 4.0, OPC-UA, indústria de moldes, digitalização do chão da fábrica, *Shop Floor Data Collection Systems*, fábricas inteligentes.

Abstract

The manufacturing industry, being one of the major boosters of the macroeconomic global development, is also an interesting investing asset for the public and private sector. Since there is a growing demand for products and services that must fulfil the needs of the global population, it becomes necessary the mass creation of products with zero waste and maximum efficiency. On a shop floor, this main goal requires solutions that allow the collection and sharing of data among several information systems that support business processes. These data aim to support operations monitoring, planning, business processes and to reduce manufacturing errors and defects. For such, it was conducted a systematic literature review using the Preferred Reporting Items for Systematic Reviews and Meta-Analyses (PRISMA) protocol to reveal in which way the industry digitalization could maximize productivity indicators. This work proposes a software architecture that serves shop floor monitoring, based on the Open Platform Communications - Unified Architecture (OPC-UA) machine-to-machine open source communication protocol, that was created by the OPC Foundation. This protocol positions itself as a de facto *standard for the industry. This work begins with a state-of-the-art that researches how the digitalization of a manufacturing enterprise can maximize integration between an information system and an Enterprise Resource Planning (ERP) and also all the services that composes the software architecture of an industrial information system, showing real time data of shop floor performance indicators and also the shop floor's manufacturing orders management digitalization. This project is part of the mobilizer project TOOLING4G – Advanced Tools for Smart Manufacturing (~ 7M€ and 32 partners) and it fits in the engineering & tooling cluster that integrates an extended value chain (from the design phase to the final product), to answer global customers' needs that more and more want turnkey solutions.*

Keywords: *Industry 4.0, manufacturing industry, shop floor digitalization, shop floor data collection systems, smart factories, OPC-UA.*

Índice

Lista de figuras

Lista de tabelas

Lista de siglas e acrónimos

API	*Application Programming Interface*
AR	*Augmented Reality*
ARM	*Acorn RISC Machine*
CoAP	*Constrained Application Protocol*
CI/CD	*Continuous Integration / Continuous Delivery*
CIA	*Confidentiality, Integrity, and Availability*
CNC	*Computer Numeric Control*
CPS	*Cyber-Physical Systems*
CRUD	*Create, Read, Update e Delete*
DCOM	*Distributed Component Object Model*
DTO	*Data Transfer Object*
EF	*Entity Framework*
ERP	*Enterprise Resource Planning*
ESTG	Escola Superior de Tecnologia e Gestão
ETL	*Extract – Transform – Load*
HTTP	*Hypertext Transfer Protocol*
HTTPS	*Hypertext Transfer Protocol Secure*
IDE	*Integrated Development Environment*
IoT	*Internet of Things*
IP	*Industrial Protocol*
IPC	*Inter-Process Communication*
Java RMI	*Java Remote Method Invocation*
JMeter CLI	*JMeter Command-line*
JSON	*JavaScript Object Notation*
KPI	*Key Performance Indicators*
MES	*Manufacturing Execution System*
MitM	*Man-in-the-Middle*
MQTT	*MQ Telemetry Transport*
MRP	*Material Requirement Planning*
MVC	*Model-View-Controller*

OEE	*Overall Equipment Effectiveness*
OFE	*Overall Factory Effectiveness*
OPC	*Open Platform Communications*
OPC-UA	*Open Platform Communications - Unified Architecture*
OPE	*Overall Plant Effectiveness*
ORM	*Object-Relational Mapping*
PEE	*Production Equipment Effectiveness*
PICO	*Population, Intervention, Comparison, Outcome*
PIN	*Personal Identification Number*
PLC	*Programmable Logic Controller*
PME	Pequenas e Médias Empresas
POO	Programação Orientada a Objetos
PRISMA	*Preferred Reporting Items for Systematic Reviews and Meta-Analyses*
PwC	*Pricewaterhouse Coopers*
RAM	*Random Access Memory*
REST	*Representional State Transfer*
RPC	*Remote Procedure Call*
SCADA	*Supervisory Control and Data Acquisition*
SFDCS	*Shop Floor Data Collection Systems*
SGBD	Sistema de Gestão de Bases de Dados
SME	*Small-Medium Enterprise*
SOA	*Service-Oriented Architecture*
SOAP	*Simple Object Access Protocol*
SPA	*Single Page Architecture*
TBF	Tempo Bruto de Funcionamento
TFN	Tempo Funcionamento Necessário
TIC	Tecnologias de Informação e Comunicação
TLF	Tempo Líquido de Funcionamento
TPM	*Total Productive Maintenance*
TRG	Taxa de Rendimento Global
TTP	Tempo Trabalho Planeado
TU	Tempo Útil

UA	*Unified Architecture*
UI	*User Interface*
UX	*User Experience*
XML	*eXtensible Markup Language*

1. Introdução

Observa-se um uso crescente das tecnologias nos produtos oferecidos e serviços prestados por todo o tipo de organizações públicas e privadas, e o setor industrial não é um caso à parte. Com o forte uso das tecnologias de informação e comunicação e também com o facto de os processos serem cada vez mais automatizados, houve a necessidade de fundir a indústria com as tecnologias para permitir troca de dados, potenciar a automação, monitorização e melhoria de processos, produtos e serviços, surgindo a quarta revolução industrial (Indústria 4.0) [1]. As revoluções que se precederam consistiram na utilização de máquinas a vapor na primeira revolução industrial (século XVIII), evoluindo para a utilização de energia elétrica na segunda revolução industrial (século XIX), culminando assim na digitalização industrial na terceira revolução industrial (século XX) [2]. Esta revolução tem como principais objetivos a descentralização das fábricas, tornando-as inteligentes ao ponto de conseguirem tomar decisões quase sem intervenção de humanos (utilizando inteligência artificial), e o fabrico orientado ao serviço, na medida em que os serviços são prestados com o auxílio de plataformas *online*, disponíveis em qualquer lugar e em qualquer momento graças à computação na nuvem. Outro objetivo sem dúvida importante que esta revolução traz consigo é a capacidade de recolher, analisar e disponibilizar os dados em tempo real, diminuindo drasticamente o tempo desde o planeamento até ao produto final tendo como vantagens, do ponto de vista operacional, os operadores ficarem mais livres de trabalhos repetitivos e de baixo valor, ou do ponto de vista socioeconómico, uma produção mais rápida e eficiente do produto, implicando assim uma redução notória do custo de fabrico que se pode evidenciar no preço final do produto e assim tornando o mercado mais acessível às massas.

Na Figura 1, é possível observar todas as quatro revoluções industriais, bem como o que estas revoluções vieram impulsionar.

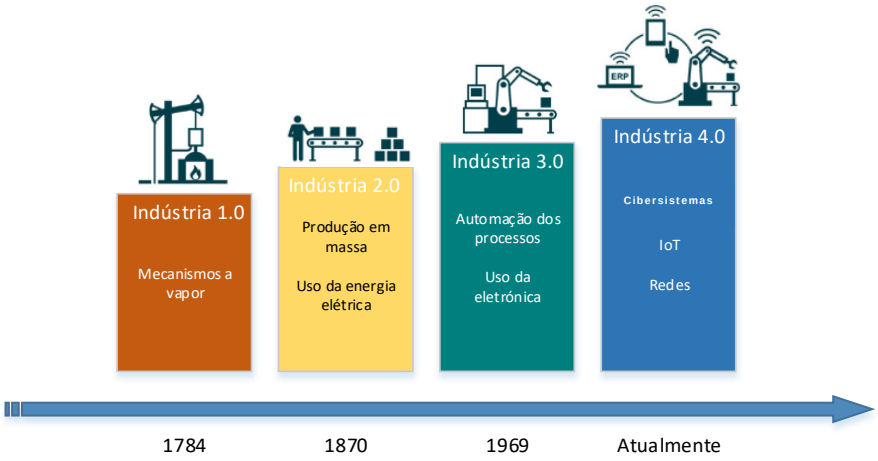

Figura 1 - Evolução Industrial (Adaptado de [7])

Contudo, as fábricas que constituem a indústria por vezes não possuem a tecnologia e o conhecimento suficientes para criar tais plataformas *on demand* necessárias para visualizar os dados, por essa razão recorrem a peritos na área para desenvolver tal tecnologia. Esta corrida das fábricas à informatização leva a que as empresas de *software* desenvolvam tecnologias proprietárias, ficando as fábricas assim dependentes dos fornecedores, limitando a sua evolução e comunicação *standard*.

A utilização de tecnologias proprietárias resulta num problema na medida em que o mercado apresenta soluções todas elas diferentes, sendo que, quando uma fábrica deseja trocar a tecnologia a ser usada, é, muitos casos, obrigada também a alterar o seu fluxo de trabalho. Para ultrapassar estes desafios surgem protocolos que ajudam no processo de recolha da informação originada das máquinas fabris, como é o caso do protocolo *Open Platform Communications* (OPC) [3], um *standard de facto* que tem como objetivo a troca de dados segura entre máquinas na automação industrial.

A tecnologia e os novos processos trazem vantagens ao nível da automação da fábrica. Potenciam o melhor conhecimento dos processos de maquinação e a troca de dados segura praticamente instantânea onde um operador tem acesso a todo o tipo de dados sobre a máquina que está a operar.

Em 2012, o conceito de Indústria 4.0 foi introduzido através de um projeto alemão com o intuito de fazer o uso do ciberespaço como uma plataforma de produção, processamento e automação [4].

Este conceito veio não só trazer novas ideias de desenvolvimento de tecnologias para uso exclusivo no espaço industrial, mas também veio a reduzir os custos nos variados segmentos da indústria e aumentar as receitas.

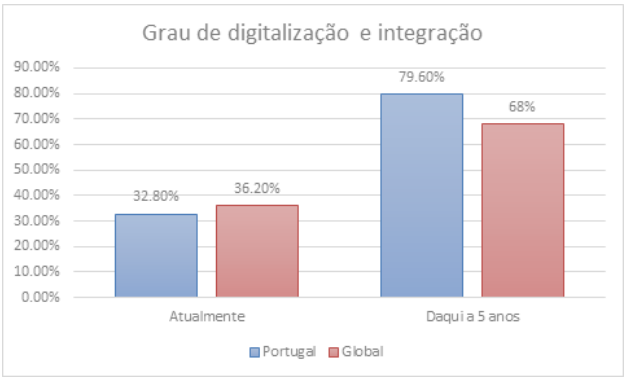

Figura 2 - Grau de digitalização e integração (adaptado de [5]**)**

Um estudo realizado pela *Pricewaterhouse Coopers* (PwC) [5] em 2016, numa época pré-COVID-19, mostra que a indústria veio a prestar mais atenção à digitalização do chão de fábrica. Segundo a Figura 2, previa-se que, em 5 anos, a nível nacional, perto de 80% das fábricas teriam forte presença no ciberespaço, com uso de aplicações e plataformas de gestão para facilitar o controlo e a troca de dados do chão de fábrica, um aumento de quase 50% face ao estado atual. Na Figura 3 é também possível observar os benefícios da digitalização industrial segundo a PwC.

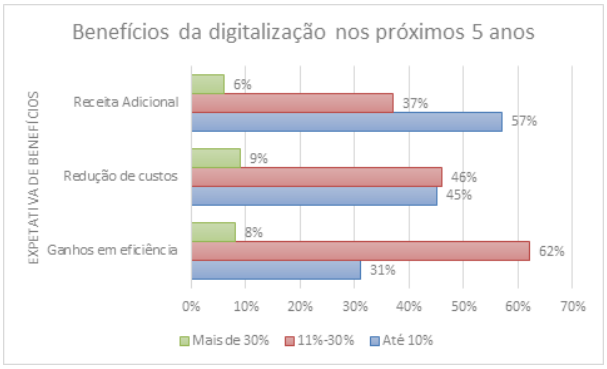

Figura 3 - Benefícios da digitalização industrial (adaptado de [5]**)**

Esta tendência não passa apenas pela simplicidade na gestão de uma fábrica, mas também traz consigo outras vantagens – redução de custos, ganhos na eficiência, disponibilidade, qualidade e lucros. Havendo um ecossistema centralizado, é possível reduzir custos, levando a uma maior eficiência da fábrica e, ao mesmo tempo, a um aumento real no lucro. O gráfico presente na Figura 3 é adaptado de um inquérito feito pela PwC em setembro de 2016 [5] às principais empresas nacionais do setor, e demonstra que até 2021, por se digitalizar uma fábrica, cerca de 62% das empresas pensam haver ganhos na eficiência de 11% a 30% enquanto que 31% das empresas pensam ter ganhos de, no máximo, 10%. Quanto à redução dos custos, cerca de 45% das empresas esperam reduzir até 10% os custos totais, enquanto que apenas 9% esperam cortar os custos acima dos 30%.

A maior parte das empresas, cerca de 57%, espera ter um aumento da receita até aos 10% enquanto que 6% espera ter receita adicional acima dos 30% em relação ao que já apresentava.

Problema

É possível entender o crescente interesse e investimento exponencial na área da Indústria 4.0 a partir deste inquérito, potencializando por sua vez mais projetos-piloto (como é o caso do projeto descrito neste trabalho), que demonstram a utilização de tecnologias emergentes em contextos práticos.

Tudo isto deve-se também ao facto de haver uma grande competição das tecnologias usadas na automação e na troca de dados, que leva ao seu baixo custo.

No entanto, uma maior digitalização da indústria coloca inúmeros desafios como, por exemplo, a grande heterogeneidade de tecnologias e protocolos, o grande volume de dados que é gerado no chão de fábrica, bem como a integração de todos os equipamentos e a aquisição de dados dos seus sensores físicos.

Objetivos

Para permitir a aquisição de dados sensoriais de um chão de fábrica, bem como a sua digitalização, armazenamento e possibilitar o processamento e visualização desses mesmos dados num formato gráfico em forma de indicadores de produtividade *Overall Equipment Effectiveness* (OEE), será realizado um levantamento de estado da arte relativo ao estado de digitalização das empresas e posteriormente será realizada a proposto um sistema de informação que permita dar suporte à digitalização de fábricas de moldes.

Para adquirir dados sensoriais de um chão de fábrica, será necessário estudar vários protocolos de comunicação que permitam a transferência de dados entre diferentes equipamentos (e.g. *Computer Numeric Control* (CNC)). Aqui, o OPC-UA apresenta-se como uma das opções que melhor se enquadra neste âmbito, podendo, a breve trecho, vir a tornar-se num *standard de facto*. Um dos desafios atuais do OPC-UA é a sua adoção de forma alargada pelos vários equipamentos presentes num chão de fábrica (com maior ou menor grau de digitalização) e, para obviar esta situação, será necessário proceder a desenvolvimentos que permitam a um operador do chão de fábrica fazer o registo de dados de uma forma manual.

Por forma a validar a arquitetura de software proposta, será realizado um estudo de caso em que se concretiza a digitalização do chão de fábrica de uma empresa do setor de moldes.

Contribuições

Espera-se com este trabalho sensibilizar para a necessidade da digitalização e sensorização na indústria, bem como para todo o processo envolvido na conceção de uma solução que vai desde a sua idealização (i.e., levantamento de requisitos funcionais e não funcionais, desenho da arquitetura), até à sua implementação e testes. Pretende-se ainda documentar e disseminar as tecnologias e protocolos que permitem a integração do grande volume de dados gerado por uma empresa no setor de moldes de modo a poder visualizar e inferir informação auxiliando assim os decisores das empresas.

O presente trabalho tem também como objetivo ser uma contribuição relevante para a área da indústria 4.0, com o estado da arte das tecnologias de comunicação e protocolos no âmbito industrial bem como os ramos técnicos mais emergentes que contribuem para a expansão de conhecimento desta mesma área.

Esta tese está dividida em capítulos, incluindo em cada um deles os seguintes conteúdos: no capítulo 2 são apresentados os métodos aplicados à revisão de sistemática da literatura e estado da arte. No capítulo 3 é apresentado um estudo de caso aplicado à digitalização de uma fábrica de moldes. No capítulo 4 é apresentada a proposta de solução para a digitalização de fábricas de moldes *Tooling 4G*, focando no processo de desenvolvimento. No capítulo 5 são discutidas as metodologias de testes que permitem verificar a assertividade desta proposta de solução no âmbito de um projeto de software. No capítulo 6 apresenta-se uma análise crítica a todo o trabalho desta tese (trabalho de investigação e de desenvolvimento) e o capítulo 7 apresenta as principais conclusões desta tese, apontando também caminhos para trabalho futuro.

2. Estado da arte

Neste trabalho foi realizado um estado da arte para identificar os desafios existentes na temática em estudo. Foram analisadas várias contribuições académicas com base nos métodos de revisão sistemática de literatura e utilizando o protocolo *Preferred Reporting Items for Systematic Reviews and Meta-Analyses* (PRISMA) [6]. Mais especificamente foram colocadas as seguintes questões de investigação científica:

- **Q1**: *Que soluções arquiteturais e tecnológicas poderão fomentar uma maior digitalização e otimização de processos de negócio de chão de fábrica?*
- **Q2**: *Como poderá a utilização de protocolos de comunicação standard contribuir para uma maior interoperabilidade entre os principais intervenientes, recursos e dados destes processos?*

De forma a responder a estas duas questões elaboradas anteriormente, foi realizado uma análise do ponto de vista do processo *Population Intervention Comparison and Outcome* (PICO). Este é um processo onde se identifica respetivamente: a população em estudo; o processo de intervenção a realizar; a comparação ou controlo que consiste em comparar duas amostragens especificas e.g. processos automatizados e processos não automatizados em fábricas e os resultados da intervenção que devem ser mensuráveis quantitativamente. A análise PICO é normalmente realizada no âmbito da especificação de objetivos na metodologia PRISMA. Neste contexto, foi realizada uma análise PICO às duas perguntas de investigação identificadas acima:

Análise PICO para a Q1:

- ***Population***: Empresas de moldes
- ***Intervention***: Aplicação de soluções tecnológicas que fomentem um maior grau de digitalização numa empresa de moldes
- ***Comparison***: A ausência de soluções tecnológicas no chão de fábrica; ou soluções de chão de fábrica que não fomentem digitalização
- ***Outcome***: Aumento no grau de digitalização; aumento de desempenho no processo de produção; aumento da eficiência de produção

Análise PICO para a Q2:

- ***Population:*** Intervenientes em processos de comunicação num chão de fábrica (e.g. sensores, máquinas, pessoas, sistemas de informação)
- ***Intervention:*** Utilização de protocolos *standard* na indústria
- ***Comparison:*** Ausência de utilização de protocolos *standard*
- ***Outcome:*** Maior interoperabilidade entre sensores, máquinas, sistemas de informação ou pessoas

2.1. Conceitos de base

Sendo que este trabalho se insere no contexto da Indústria 4.0 e sendo esta uma área multidisciplinar, julga-se oportuno a apresentação de alguns conceitos base referentes a algumas tecnologias que compõe o universo da Indústria 4.0 que permita ao leitor a aquisição de algum contexto relativamente ao estado da arte apresentado no próximo capítulo.

2.1.1. Visão computacional

David Marr em 1982 [7], no seu livro *Vision* proporcionou uma análise do ponto de vista da neurociência que contribuiu para um entendimento de como o cérebro humano consegue diferenciar formas geométricas e como funciona o processamento de informação visual, sendo este considerado um dos berços de uma das áreas mais relevantes dentro da Indústria 4.0 – a visão computacional.

A visão computacional, sendo um ramo de investigação abrangente, foi também responsável pelo aparecimento de outras áreas de investigação mais específicas, nomeadamente o da realidade aumentada (*Augmented Reality* – AR).

O aparecimento deste novo ramo de investigação, possibilitou a muitos engenheiros e investigadores a criação de novas tecnologias que visassem a solução de problemas em áreas que não eram necessariamente adjacentes à de visão computacional.

2.1.2. Sistemas ciber-físicos

A indústria 4.0 trouxe vários conceitos novos, sendo um deles os *Cyber-Physical Systems* (CPS). Os CPS são sistemas que geram modelos que se adaptam a alterações físicas numa fábrica, adquirindo dados através de outras aplicações (e.g. aplicações de monitoramento do chão de fábrica, monitoramento de sensores físicos de máquinas industriais, etc.) [8]

Os CPS são uma nova geração de sistemas que permitem a integração do mundo real com as capacidades computacionais. A partir de tecnologias como a *Internet of Things* (IoT), é possível a um CPS o acionamento de ações com base na troca de dados entre componentes de uma forma autónoma [9,10] (comunicação *Machine to Machine* (M2M)). Contextualizando num ambiente de produção, os CPS implicam que exista comunicação automatizada entre todos os níveis de produção – desde os sensores físicos até redes de logística [9,10]. Para atingir este nível de autonomia a interoperabilidade entre objetos físicos e virtuais torna-se um requisito fundamental, sendo que, para atingir esse objetivo, é necessário o desenvolvimento de *standards* e protocolos que permitam *interfaces* de comunicação viáveis, que sejam padronizados no ambiente industrial e que também suportem todas as implicações e exigência de um ambiente de produção, e.g., segurança no transporte de dados, tolerância a falhas, entre outros.

2.2. Revisão sistemática da literatura

Uma revisão sistemática de literatura foi realizada para se tentar perceber se a digitalização de uma empresa na indústria de moldes poderia, como consequência, possuir vantagens ao nível do desempenho do processo de produção. Como tal, algumas bases de dados eletrónicas como o *Google Scholar* e o *Scopus* foram utilizadas para obter artigos que ajudassem a respondessem a esta questão. Estes repositórios foram escolhidos pela sua abrangência, facilidade de acesso e as restrições de tempo associadas a um trabalho desta natureza. Os repositórios eletrónicos foram utilizados entre os meses de dezembro de 2019 e janeiro de 2020.

As *keywords* utilizadas nestas duas bases de dados e respetivos resultados das pesquisas são apresentados na Tabela 1.

Tabela 1 - *Keywords* utilizadas e número de resultados

Google Scholar	
industry 4.0 manual processes	559,000 resultados
industry 4.0 key performance indicators	482,000 resultados
industry 4.0 OEE	16,900 resultados
Scopus	
industry 4.0 AND performance	1,650 resultados
advantages AND industry 4.0	625 resultados

Aplicando a metodologia PRISMA, foi possível realizar uma seleção dos artigos a serem analisados no âmbito do projeto. Um dos meios utilizados para se proceder à seleção foi incluindo critérios de inclusão e de exclusão para um os artigos identificados.

Tabela 2 - Descrição dos critérios de exclusão

Critério	Descrição
Search engine reason/Motor de pesquisa (MP)	Se não for possível aceder ao artigo devido ao motor de pesquisa.
Without full text/Sem full text (SFT)	Não é possível conseguir a versão *full text* de um artigo.
Non relevant/Não relevante (NR)	O conteúdo de um artigo não aborda os pontos das questões de investigação.
Loosely-related/Fracamente relacionado (FR)	O conteúdo do artigo aborda os pontos das questões de investigação, mas de uma forma descontextualizada.

Tabela 3 - Descrição dos critérios de aceitação

Critério	Descrição
Partially-related/Parcialmente relacionado (PR)	O conteúdo do artigo aborda pontos das questões de investigação, mas não de uma forma sucinta ou não as tem como assunto principal.
Closely-related/Intimamente relacionado (IR)	O conteúdo do artigo aborda exatamente pontos das questões de investigação ou é uma resposta às mesmas.

A Tabela 2 e a Tabela 3 representam respetivamente os critérios de exclusão e os critérios de aceitação que foram utilizados para selecionar ou excluir um artigo.

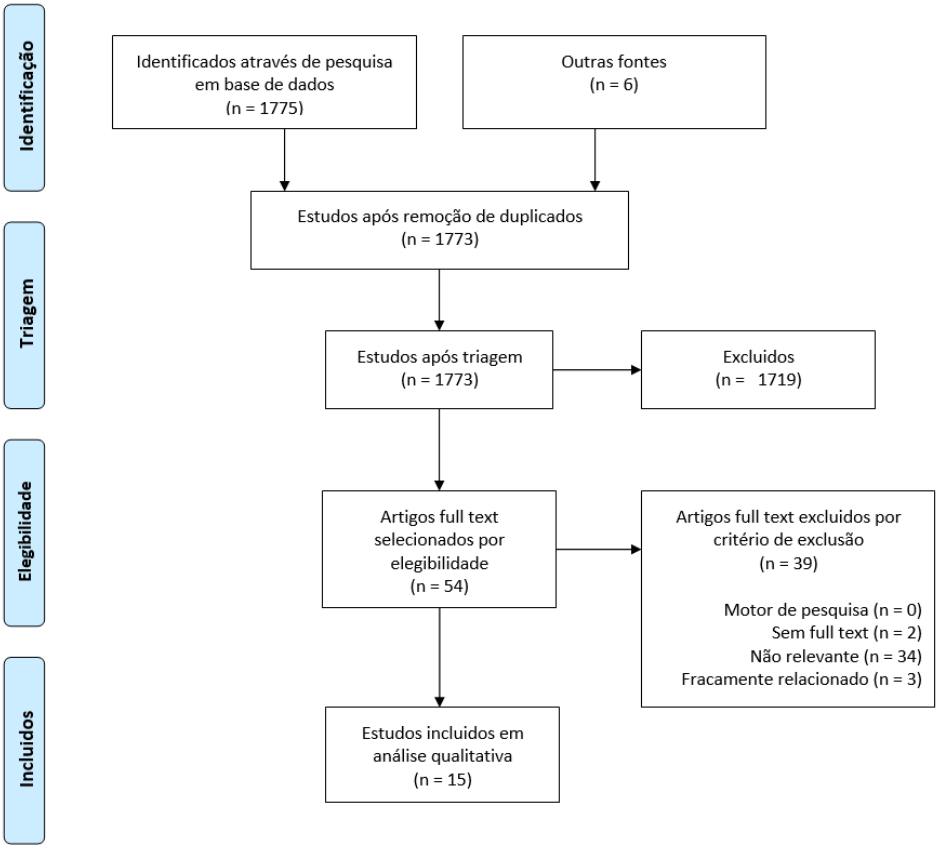

Figura 4 - Diagrama PRISMA flow

Aplicando a metodologia PRISMA, como observado na Figura 4, consegue-se entender o processo iterativo de investigação utilizado neste âmbito. Na fase de identificação, são identificadas *keywords* pertinentes ao tema e que vão de encontro às questões de investigação acima referidas (Tabela 1), neste caso foram identificados 1775 artigos em bases de dados e 6 registos noutras fontes. Procede-se ao processo de triagem, onde o objetivo é eliminar artigos duplicados, indisponíveis ou que não vão de todo ao encontro do que se pretende.

Nesta fase foram eliminados 1719 artigos, tendo sobrado 54 que foram posteriormente consultados para a fase de elegibilidade. Na fase de elegibilidade, são aplicados os critérios de aceitação e exclusão, definidos anteriormente (Tabela 2, Tabela 3), do qual resultaram 15 artigos selecionados para análise qualitativa.

No contexto de digitalização de uma fábrica de moldes, foi constatado que existem vários ramos de investigação referentes que também dizem respeito à temática da Indústria 4.0. Dentre os quais, nomeadamente – visão computacional, sendo a realidade aumentada um tópico bastante debatido; os sistemas de informação empresariais (i.e. *Enterprise Resource Planning* – ERP, *Manufacturing Execution System* – MES, etc.); os sistemas ciber-físicos (*Cyber-Physical Systems*); a otimização do processo de produção com base na monitorização de indicadores de desempenho, como os OEE.

Em 2003, *Tang et al* [11] realizaram um estudo em que compararam a utilização de realidade aumentada com a não utilização da mesma, na montagem de objetos numa fábrica, ao qual concluíram que a utilização de realidade aumentada estava fortemente correlacionada com um aumento do desempenho na execução de tarefas bem como no alívio do *stress* mental nas tarefas de montagem por parte dos operários fabris.

Em 2017, *Ras et al* [12] criaram um *roadmap*, já no contexto da Indústria 4.0 que já previa a utilização de *Cyber-Physical Systems* (CPS) e *Augmented Reality* (AR). Também referem os principais desafios da área – *Big Data* pelo grande volume de dados gerado pelos CPS; a utilização eficiente desse *Big Data* para aperfeiçoar o trabalho realizado por humanos e máquinas; a complexidade dos sistemas em configurações, ao serem operados e também na sua manutenção.

Prevê-se que a visão computacional e a realidade aumentada serão duas ferramentas indispensáveis no contexto da indústria 4.0. Não só pelo aumento de eficiência que é proporcionado aos seus utilizadores (i.e. operadores fabris), mas também por permitirem um menor *stress* cognitivo. Não obstante, as utilizações destas duas vertentes de investigação também criaram ramos especializados de investigação e também novos problemas, sendo elas tecnologias que ainda precisam de maturação em âmbito de produção industrial.

Em 2017, *Bauman et al* [13], implementaram um CPS para fazer a aquisição de dados de impressora 3D e os seus sensores físicos associados. Para facilitar a comunicação *Machine-to-Machine* (M2M) entre os sensores, um *Raspberry Pi 2* e a impressora 3D e também o acesso seguro dos dados a partir da Internet. Foi utilizado o protocolo OPC-UA devido à sua interoperabilidade.

Em 2019, *Lam N. A. E et Haugen O.* [10] desenvolveram um projeto de investigação que consistia na implementação de um CPS utilizando o modelo arquitetural *Service-Oriented Architecture* (SOA).

A arquitetura tinha de suportar requisitos como a aquisição de dados de sensores físicos, segurança de comunicação, *service registry* bem como visualização dos dados. De forma a atingir os objetivos propostos, os autores utilizaram o protocolo *Open Platform Communications – Unified Architecture* (OPC-UA) que permitiu a interoperabilidade na comunicação, a segurança no transporte dos dados e também a autorização no acesso à informação através da *framework Arrowhead*.

2.3. Protocolos para comunicação de sensores físicos em contexto industrial

Como referido na secção 2.1.2 (Sistemas ciber-físicos), os CPS permitem a integração do mundo real com as capacidades computacionais. Num contexto industrial, a aquisição de grandezas físicas como humidade, pressão, temperatura e outras permitem extrapolar o desempenho do processo de produção. As corretas agregações destes dados permitem a sua utilização enquanto indicadores, para uma previsão rigorosa de quando, por exemplo, uma máquina pode estar prestes a falhar. Sendo estes dados de suma importância, torna-se impreterível o estudo de soluções que permitam a transmissão destes dados por sistemas industriais de uma forma eficiente e segura.

Com o início da terceira revolução industrial, por volta de 1970, a utilização da robótica colaborativa e da digitalização dos processos industriais mostrou-se uma vantagem para a eficiência dos processos produtivos. Como tal, iniciou-se a investigação e desenvolvimento de protocolos que permitissem a transmissão de dados, que por essa época, eram na sua generalidade, originadas por *Programmable Logic Controller* (PLC) em sistemas *Supervisory Control and Data Acquisition* (SCADA).

2.3.1. Modbus

Vários protocolos foram produzidos para sistemas SCADA, sendo muitos baseados no protocolo Modbus [14]. Protocolo original desenvolvido em 1979, as redes Modbus eram isoladas e eram redes livres de ameaças de segurança, logo requisitos tais como integridade, autenticação e não repúdio não tinham sido tidos em consideração no desenvolvimento deste protocolo [15].

2.3.2. Profibus

Desenvolvido em 1987, é um protocolo que utiliza o modelo assimétrico *master-slave* para o acesso ao canal de informação. O Profibus utiliza um *token* para permitir às estações *master*, o acesso ao barramento, permitindo assim comunicar com estações *slave* [16].

2.3.3. EtherNet/IP

Desenvolvido em 1995, EtherNet/IP [17] é um protocolo que adapta o standard *Ethernet* no contexto industrial (*Industrial Protocol* – IP). Utiliza uma arquitetura *full-duplex* que permite diminuir a latência na comunicação, bem como a colisão entre pacotes. Possui desempenho em tempo real, sendo que um dos seus maiores contras é não ser estritamente determinístico [18].

2.3.4. OPC

Com o aparecimento de ataques cibernéticos a grandes indústrias como o caso da *Stuxnet*, que consistiu num *worm* que infetou o sistema de controlo das plantas nucleares e configurou as centrifugadoras para velocidades altas, causando o sobreaquecimento e a consequente explosão das plantas de enriquecimento de urânio, a cibersegurança num contexto industrial ganhou outra importância [19]. O protocolo OPC é um exemplo de um protocolo industrial que permite a segurança na comunicação. Desenvolvido em 1996, utiliza *Remote Procedure Call* (RPC) e *Distributed Component Object Model* (DCOM) para permitir comunicação de processos em tempo real numa rede *Ethernet* com um modelo cliente-servidor [18].

2.3.5. OPC-UA

Sendo o protocolo OPC, um protocolo bastante fragmentado em termos de variantes do protocolo (i.e. OPC *Data Access*, OPC *Historical Data Access*, OPC *Data Exchange*, etc.), surgiu a necessidade de unir todas as variantes num só protocolo, surgindo assim, em 2006, o protocolo OPC-UA (*Unified Architecture* – UA). A nova versão do protocolo OPC vem eliminar a dependência no sistema operativo Windows, com camadas adicionais de segurança e também uma arquitetura baseada em *Service-Oriented Architecture* (SOA).

Em 2016, *Schleipen et al* [20] mostraram como o OPC-UA pode ser utilizado para soluções que exigem elevado desempenho em ambiente industrial sendo que as suas características se enquadram nos princípios da Indústria 4.0.

Grüner et al [21] em 2016 mostraram que é possível a integração do protocolo OPC-UA com o modelo arquitetural *Representional State Transfer* (REST) para encapsular as funções proporcionadas pelo OPC-UA e conseguindo manter o desempenho e *real-time*.

Müller et al [22] em 2017 desenvolveram uma solução que permitisse a leitura de um sensor físico de temperatura proveniente de uma impressora 3D, utilizando para efeito um *Arduino Yun* com um servidor OPC-UA modificado para o efeito, mostrando que este protocolo também se enquadra em soluções para sistemas embebidos.

Ainda em 2017, *Derhamy et al* [23] desenvolveram investigação num campo que ainda se considerava aberto – a interoperabilidade entre o protocolo OPC-UA e outros protocolos de comunicação. Como tal, utilizaram o modelo "*Service Translator*" proposto pelo projeto *Arrowhead* com um mapeamento para um formato intermédio e que poderia ser utilizado em conjunto com outros protocolos como *Constrained Application Protocol* (CoAP), *Hypertext Trasnfer Protocol* (HTTP) e *MQ Telemetry Transport* (MQTT).

Em 2019, *Lam* e *Haugen* [10], conceberam um sistema CPS para monitorização de dados de sensores utilizando o protocolo OPC-UA com uma arquitetura bastante escalável e *cloud based* preparado para a implementação de sensores agnósticos.

2.4. Resultados

Neste subcapítulo sintetiza-se todos os artigos científicos revistos no âmbito da revisão sistemática desenvolvida em forma tabular, enaltecendo uma pequena conclusão de cada artigo, a sua data de publicação, bem como a linha de investigação. A ordenação da tabela está feita pela linha de investigação.

Tabela 4 - Artigos revistos para revisão sistemática

Nome do artigo	Data	Autores	Linha de investigação	Conclusões
The impact of the fourth industrial revolution: a cross-country/region comparison	2018	*Liao, Y. et al.*	Digitalização nas pequenas e médias indústrias	Enfatiza alguns dos que são os pontos fortes da digitalização da indústria comparativamente com um processo maioritariamente manual.
Comparative Effectiveness of Augmented Reality in Object Assembly	2003	*Tang, A. et al.*	Realidade Aumentada	É realizado um estudo que evidencia as vantagens da Realidade Aumentada no chão de fábrica.
Bridging the Skills Gap of Workers in Industry 4.0 by Human Performance Augmentation Tools – Challenges and Roadmap	2017	*Ras, E. et al.*	Realidade Aumentada	Os autores propõem os quatro principais desafios impostos pela utilização de tecnologias de Realidade Aumentada.
High Performance Manufacturing – An Innovative Contribution towards Industry 4.0	2016	*Sandengen, O. et al.*	Indicadores OEE	Revela de que forma a utilização de princípios da indústria 4.0, bem como indicadores de desempenho (OEE) auxiliam a eficiência no processo de fabricação aditiva.
A critical review of smart manufacturing & Industry 4.0 maturity models: Implications for small and medium-sized enterprises (SMEs)	2018	*Mittal, S. et al.*	Digitalização nas pequenas e médias indústrias	Refere os motivos pelos quais é difícil a uma *Small-Medium Enterprise* - SME) adotar a Indústria 4.0, mesmo com todas as suas vantagens competitivas de mercado.
An Empirical Investigation of the Relationship between Overall Equipment Efficiency (OEE) and Manufacturing Sustainability in Industry 4.0 with Time Study Approach	2018	*Yazdi, P. et al.*	Indicadores OEE	Projeto de investigação em que se estuda a utilização e otimização de OEE no contexto da sustentabilidade de produção, utilizando um algoritmo para o efeito.

Assembly system configuration though Industry 4.0 principles: the expected change in the actual paradigms	2017	*Cohen, Y. et al.*	Digitalização de chão de fábrica	Digitalização de um chão de fábrica com a implementação de inteligência entre máquinas e criação de uma aplicação para os operadores utilizarem no chão de fábrica.
Ubiquitous knowledge empowers the Smart Factory: The impacts of a Service-oriented Digital Twin on enterprises' performance	2019	*Longo, F. et al.*	Digitalização de chão de fábrica	Digitalização de um chão de fábrica, permitindo a utilização de ferramentas informáticas pelos colaboradores bem como a utilização de *dashboards.*
OPC UA & Industrie 4.0 - enabling technology with high diversity and variability	2016	*Schleipen, M. et al.*	Protocolos de comunicação industriais	Refere vantagens na utilização do protocolo OPC-UA no contexto de chão de fábrica.
RESTful Industrial Communication With OPC UA	2016	*Grüner, S. et al.*	Protocolos de comunicação industriais	Refere a possível coexistência e compatibilidade entre o protocolo OPC-UA e o *standard* REST.
Developing Open Source Cyber-Physical Systems for Service-Oriented Architectures Using OPC UA	2017	*Müller, M. et al.*	Protocolos de comunicação industriais	Projeto de investigação que utiliza um Arduino Yun com um servidor costumizado OPC-UA para medir a temperatura de um sensor proveniente de uma impressora 3D.
A Web-based Platform for OPC UA integration in IIoT environment	2017	*Cavalieri, S. et al.*	Protocolos de comunicação industriais	Demonstra a utilização de um *dashboard* que monitoriza sensores em tempo real utilizando o protocolo OPC-UA.
Cyber-physical System Control via Industrial Protocol OPC UA	2017	*Baumann, F. et al.*	Protocolos de comunicação industriais	Projeto de investigação que consiste na criação de um sistema ciber-físico para monitoramento de sensores via OPC-UA.
Implementing OPC-UA services for Industrial Cyber-Physical Systems in Service-Oriented Architecture	2019	*Lam, A. Haugen, O.*	Protocolos de comunicação industriais	Criação de um sistema CPS para monitoramento de dados de sensores via OPC-UA com uma arquitetura altamente escalável e preparada para integração agnóstica com dados de sensores físicos.
Protocol interoperability of OPC UA in Service Oriented Architectures	2017	*Derhamy, H. et al.*	Protocolos de comunicação industriais	Aborda a interoperabilidade entre o protocolo OPC-UA e outros protocolos.

2.5.Discussão e conclusão

O objetivo desta revisão sistemática de literatura foi demonstrar o panorama global das mais recentes tecnologias empregues no contexto da Indústria 4.0 e estudar de que forma as microempresas, as pequenas e médias empresas bem como as empresas bem estabelecidas na indústria poderiam tirar proveito das técnicas que são empregues na digitalização de um chão de fábrica.

Como é expetável, as empresas mais pequenas têm uma maior correlação com um grau de digitalização menor, ou seja, têm empregado nos seus processos de produção metodologias maioritariamente manuais ou pouco automáticas, enquanto que as empresas maiores têm um maior grau de digitalização. Este fenómeno está diretamente relacionado também ao poder de compra da empresa em si, sendo que uma empresa maior tem uma maior capacidade de produção e consequentemente de gerar maiores mais valias. As maiores empresas também têm uma maior tendência em investir em investigação e desenvolvimento e na otimização dos seus processos de produção, o que impulsiona a implementação de tecnologias e metodologias da Indústria 4.0 nos seus processos.

No âmbito tecnológico da digitalização do chão de fábrica, vários modelos de comunicação e protocolos foram estudados nesta revisão sistemática de literatura. Foi apurado que, principalmente nos Estados Unidos da América, existe uma heterogeneidade de protocolos a serem utilizados, mas também existe uma predominância em três protocolos (i.e. Modbus, Ethernet/IP e OPC/OPC-UA) e uma tendência em adoção do OPC-UA principalmente devido a vantagens ao nível de integração dos dados com a *Cloud*, IoT e *Big Data*.

Como referido, neste estudo foi incluído como *target* tanto empresas de moldes de grande dimensão como de pequena e uma das limitações deste estudo foi exatamente a pouca quantidade de estudos que referissem o estado da digitalização das empresas de moldes por país, para se conseguir inferir quantitativamente as vantagens da adoção de tecnologias da Indústria 4.0. Outra desvantagem é referente ao grau de adoção da Indústria 4.0 - sendo este um conceito relativamente novo, é expectável que só os países mais desenvolvidos (principalmente na área da indústria) tenham mais investigação desenvolvida nesta área, o que limita o universo de artigos analisados a países como a Alemanha, os Estados Unidos da América ou a China.

Seria interessante implementar outros métodos de investigação aliados à revisão sistemática de estado da arte realizada (e.g. entrevistas) a outros países, como por exemplo Portugal, para se poder extrapolar se os resultados demonstrados neste estudo se representam também noutros países.

Também não foi possível encontrar na literatura soluções estado da arte que fizessem a integração entre o novo protocolo OPC-UA com processos de digitalização maioritariamente manuais, que pudessem ser posteriormente processadas por sistemas de logística e gestão e.g. *Enterprise Resource Planning* (ERP) e *Manufacturing Execution Systems* (MES). Sendo que estes sistemas de informação estão intrínsecos nos processos de produção, seria bastante interessante observar de que forma a integração dos processos manuais de um chão de fábrica com os processos de gestão realizados por exemplo num ERP poderiam funcionar utilizando um protocolo como o OPC-UA.

Este trabalho de investigação focou-se no estudo das vantagens da digitalização de um chão de fábrica, sendo que nos capítulos seguintes será discutido um caso de estudo referente a digitalização de um chão de fábrica de uma fábrica de moldes.

Como linhas de investigação futuras, seria interessante estudar de que forma a aquisição inteligente de dados brutos gerados por um chão de fábrica poderiam auxiliar processos de decisão e ao mesmo tempo fomentar uma maior eficiência que permitisse um aumento dos indicadores de desempenho.

3. Estudo de caso - digitalização de chão de fábrica de uma empresa do setor industrial dos moldes – Análise concetual

Para validar os requisitos da arquitetura de software a propor no capítulo seguinte, realizou-se um estudo de caso numa empresa de moldes.

Tipicamente, o processo de produção numa fábrica começa com a criação de uma ou várias ordens de fabrico/produção. Estas são enviadas para o chão de fábrica e o material necessário para a execução da ordem de fabrico é requisitado. No caso de não existir *stock* de algum material, é criado um pedido para reposição do mesmo e os recibos de compra de material são guardados.

Durante o processo de fabrico do produto ou dos produtos, as horas de trabalho dos funcionários são registadas e associadas a um posto de trabalho e a uma ou mais ordens de fabrico.

No final do processo, é criado um registo onde constam as peças produzidas corretamente e com falha. Estes registos podem ser utilizados para relatórios e também podem ser utilizados para gerar indicadores de produtividade (i.e. *Key Performance Indicators* – KPI).

O processo de execução de uma ordem de fabrico acima descrito pode ser manual, automatizado ou um híbrido entre os dois, dependendo do estado de digitalização de uma fábrica, bem como das suas necessidades de produção e capacidades financeiras.

O grande desafio relativamente à digitalização de um cenário como o acima descrito passa pela integração de equipamentos que podem não ter capacidade de computação, o que implica o registo manual das ordens de fabrico. Podem existir postos de trabalho sem qualquer tipo de sensorização, ou com sensorização limitada, o que implica a criação de soluções *ad-hoc* para a uniformidade e normalização dos dados enviados, bem como dos protocolos de comunicação que permitam a transferência de dados seguros num ambiente *real time* e por fim capacidades de processamento e armazenamento de grandes volumes de dados, i.e., várias máquinas a gerar dados em intervalos de tempo curtos e em tempo real.

Outro grande desafio prende-se com a possibilidade de se poder adicionar capacidade sensorial a um equipamento (i.e., um novo sensor físico), procedendo para o efeito a alterações mínimas ou nulas na solução arquitetural de software a desenvolver (habilitando, assim, interoperabilidade semântica sobre os dados gerados por sensores físicos).

Com isto, um dos requisitos deste projeto foi o da digitalização de parte deste processo. Utilizando um sistema de informação do tipo *Enterprise Resource Planning* (ERP) como repositório centralizado de todas as ordens de fabrico, foi proposta uma solução arquitetural e tecnológica que permite a aquisição das ordens de fabrico provenientes de um ERP mantendo os princípios da interoperabilidade, por forma a permitir que a solução fosse adaptada a diversas situações, em casos de estudo diferentes exigindo para o efeito o mínimo de alterações possíveis para funcionar com outros ERP. Construiu-se, para o efeito, um protótipo funcional de uma aplicação que permite a autenticação de um utilizador para a execução de ordens de fabrico, a criação de equipas, e a execução de uma ordem de fabrico, sendo que a esta podem estar afetas uma ou mais pessoas. É também possível visualizar todas as ordens de fabrico por serem executadas e selecionar uma. Após a execução da ordem de fabrico, o operador terá de inserir o número de unidades bem-sucedidas e com defeito, sendo que no final do processo, é enviado para o ERP um registo com estes mesmos dados.

Também de forma a possibilitar uma eficiente tomada de decisão pelos decisores do chão de fábrica, foi criado um *dashboard* que apresenta em formato de gráfico os indicadores de desempenho gerados a partir dos dados transmitidos pelas máquinas e sensores do chão de fábrica. A criação deste *dashboard* prende-se com a necessidade de percecionar quando um equipamento ou uma bancada não se encontra a trabalhar na sua capacidade máxima. Desta forma é possível prever também falhas eminentes no processo de produção.

Com a heterogeneidade de protocolos de comunicação e *standards* utilizados num chão de fábrica por diferentes equipamentos de produção e equipamentos de sensorização, pensou-se também na uniformização deste cenário com um único protocolo de comunicação *standard* na indústria (que alguns equipamentos mais avançados já possuem). No entanto, verificou-se, neste estudo de caso, a existência de bastantes equipamentos sem capacidades de computação, transmissão de dados, ou que não possuem compatibilidade com este mesmo protocolo de comunicação. Para colmatar esta limitação, idealizou-se também a implementação de uma solução tecnológica *ad-hoc* deste mesmo protocolo, a ser utilizada por estes equipamentos.

Para dar resposta a muitas das necessidades da indústria tais como, entre outras, a integração entre sistemas de informação de gestão (e.g. ERP, MES), foi pensada uma solução que pudesse ser utilizada em várias empresas na indústria de moldes. Desta forma, será descrito o processo de desenvolvimento da solução no contexto de um projeto de *software*. Proposta de solução para digitalização de um chão de fábrica

Tendo o estudo realizado em âmbito de estado da arte evidenciado que a utilização de sistemas de informação que agreguem os dados de um chão de fábrica é uma prática que tem sido corrente, com a consequência de fomentar um aumento na produtividade, bem como na eficiência no processo de produção, foi realizado neste âmbito o desenvolvimento de uma proposta de solução arquitetural e tecnológica para a digitalização do chão de fábrica. Sendo este um projeto de desenvolvimento de engenharia de software, nos subcapítulos seguintes é descrito o processo de desenvolvimento desta mesma proposta.

3.1. Análise de requisitos

No âmbito deste projeto, foi criado um consórcio entre várias entidades que participaram neste projeto que vão desde empresas da área de moldes, *software houses* e também instituições académicas. Foram realizadas reuniões de consórcio e foi documentado uma lista de requisitos funcionais e não funcionais. Sendo este um projeto multidisciplinar, só serão descritos os requisitos inerentes ao desenvolvimento realizado.

3.1.1. Requisitos Funcionais

RF1	A solução deve apresentar um *dashboard* que permita a visualização de indicadores *Overall Equipment Effectiveness* (OEE)
RF2	Deve ser possível poder filtrar os OEE por equipamento e por data
RF3	A solução deve permitir o *Create, Read, Update e Delete* (CRUD) de operadores
RF4	Um operador no chão de fábrica deve poder registar uma ordem de trabalho
RF5	Um operador deve poder criar uma equipa
RF6	Um operador e a sua equipa devem poder escolher a sua ordem de trabalho
RF7	Um operador deve poder ordenar todas as ordens de trabalho a realizar por data de início
RF8	Um operador deve poder filtrar todas as ordens de trabalho a realizar

RF9	Um operador, após terminar a realização de uma ordem de trabalho, deve poder dar a contagem de peças realizadas com sucesso / com falha
RF10	Todas as ordens de trabalho devem ser registadas no ERP
RF11	A solução deve permitir o registo de dados de sensores provenientes de equipamentos para o cálculo de OEE

3.1.2. Requisitos Não Funcionais

RNF1	A solução deve ser agnóstica ao ERP
RNF2	A integração com novos ERP deve ser feita com o mínimo de esforço de implementação possível
RNF3	A visualização dos dados referentes a indicadores operacionais/desempenho só pode ser realizada com autenticação e autorização
RNF4	A solução deve permitir uma gestão universal dos dados dos sensores, independentemente do tipo de sensor
RNF5	A aplicação que permite a recolha dos dados de produção no chão de fábrica deve ser responsiva
RNF6	A aplicação que permite a recolha dos dados de produção no chão de fábrica deve poder ser utilizada em diferentes sistemas operativos
RNF7	O sistema deve ser escalável de forma a permitir que com o mínimo de esforço de implementação se adicione outro módulo
RNF8	O sistema deve permitir uma utilização intuitiva e fácil
RNF9	O sistema deve permitir 100 utilizadores em simultâneo com tempo de resposta inferiores a 5 segundos
RNF10	O sistema deve permitir a fácil e rápida manutenção de código

3.2. Arquitetura da solução

Com base nos requisitos não funcionais apresentados, foi escolhido o padrão arquitetural *Service Oriented Architecture* (SOA) para permitir a uma solução ser mais facilmente testável graças a uma maior separação de responsabilidades entre cada um dos serviços (princípio da separação de responsabilidades). Desta forma também é possível manter uma escalabilidade independente por serviço – é expetável que existam módulos com uma maior necessidade de processamento bem como de armazenamento, e com o SOA, não existe a necessidade de encarecer muito mais os custos de infraestrutura da solução (i.e., em comparação com a utilização de uma solução completamente monolítica).

Muitos dos requisitos funcionais não tinham uma dependência direta entre si e, criando módulos específicos para requisitos específicos, também se permite que, no caso de um serviço perder a disponibilidade, a solução como um todo não é afetada.

Na Figura 5, é apresentado um diagrama arquitetural simplificado idealizado como proposta de solução para este problema.

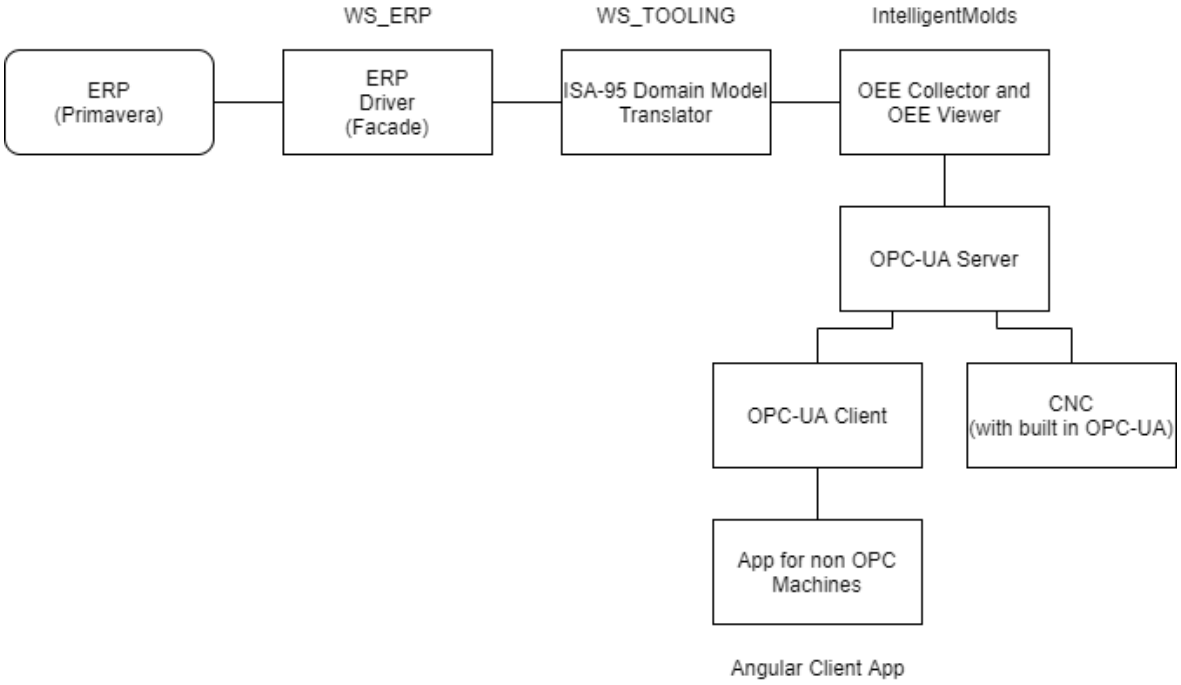

Figura 5 - Diagrama arquitetural simplificado Tooling 4G

Este diagrama (Figura 5) revela os diferentes componentes do idealizado sistema de informação *Tooling 4G*. Cada bloco representa um componente isolado, sendo o sistema de informação composto pela integração destes diferentes componentes isolados com funções especificas. Do diagrama simplificado apresentado anteriormente, resultou um diagrama *Unified Modeling Language* (UML) de *deployment* representativo dos módulos presentes na solução proposta (Figura 6).

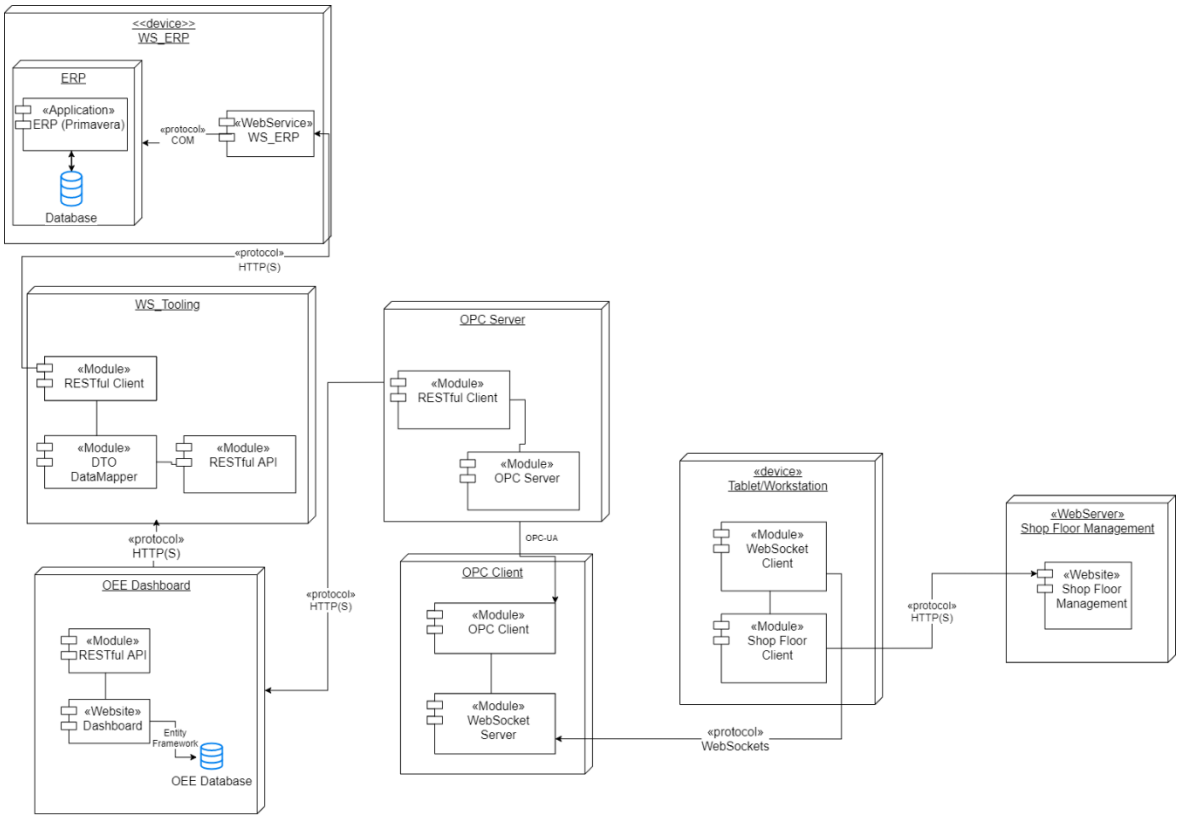

Figura 6 - Diagrama UML de *deployment* da solução.

Seguindo o padrão arquitetural SOA, obteve-se uma solução em que cada bloco do diagrama corresponde a um serviço. Cada serviço corre num processo isolado, não havendo partilha de memória, sendo atribuído a cada um, o seu próprio *address space*.

Outros exemplos destas tecnologias num contexto mais recente é o *Simple Object Access Protocol* (SOAP) e o *Representational State Transfer (*REST).

O REST foi o mecanismo de *Remote Procedure Call* (RPC) escolhido, pelo facto de ser já um *standard* bastante aceite na indústria, bem como devido à sua simplicidade e apenas depender do protocolo HTTP para a sua comunicação. O facto do REST não possuir estados (bem como o protocolo HTTP), torna este protocolo mais simples de ser utilizado e mais escalável [24]. É também mais simples a hospedagem de um serviço REST, sendo que este ao basear-se em HTTP, tudo o que é necessário é um servidor HTTP, que tem sido o *standard de facto* desde o início da era da Internet.

Desta forma, optou-se então por utilizar o *standard* REST na comunicação nos serviços WS_ERP, WS_Tooling, OEE Web Server, sendo que os restantes tinham outras necessidades de comunicação que serão descritas mais detalhadamente nos subcapítulos seguintes.

3.3.Modelos de dados

Neste capítulo serão abordados os modelos de dados e de metadados que foram utilizados no âmbito deste projeto. Mais especificamente, será descrita a implementação do standard ISA-95 que foi utilizado neste projeto para representação dos *Data Transfer Object* (DTO), bem como será discutido o modelo de dados que foi idealizado para persistência da informação (em meios persistentes – base de dados).

Um *Data Transfer Object* (DTO) é um padrão de desenho, utilizado normalmente em engenharia de software no âmbito de transferência de dados entre dois sistemas ou subsistemas diferentes [25]. Tecnicamente, consiste numa estrutura de dados que seja representativa de uma entidade do modelo de dados, i.e., uma classe, que seja serializável – que possa ser transformada num formato transferível ou armazenável. As vantagens técnicas referentes à utilização deste padrão de desenho prendem-se com a eliminação de uma parametrização customizada e de tamanho extenso, que teria de ser realizada na sua ausência, ou seja – torna-se desnecessário o processo tedioso e com tendencioso a falhas de passar cada parâmetro que se quer transferir individualmente entre sistemas, e envia-se uma "estrutura de dados" organizada como alternativa.

Mais informação relativamente ao standard ISA-95 poderá ser encontrada no capítulo 4.2.3.

Modelo de dados para representação de DTO

De forma a que a comunicação dos dados seja *standard*, o que auxilia também a comunicação com os *Material Requirement Planning* (MRP), sendo que muitos se baseiam nestes *standards* para a representação agnóstica de dados, optou-se por também caraterizar os atributos das classes dos dados que provêm de ERP/MRP com base na norma ISA-95 parte 1 e parte 2.

O *standard* ANSI/ISA-95 [26] é um *standard* internacional que permite criar uma "ponte" para transmissão de dados entre os sistemas de alto nível (e.g. ERP, MES, *etc*.) e sistemas de informação utilizados no chão de fábrica (i.e., sistemas de gestão de chão de fábrica).

Existiu um esforço de tentar adaptar ao máximo o modelo de dados ao ISA-95, sendo que nem todos os atributos foram utilizados. Em algumas classes (*e.g. Company*) não existia meta dados correspondentes no ISA-95, o que obrigou a que boa parte do modelo de dados fosse *ad-hoc* (baseado no trabalho que já tinha sido desenvolvido anteriormente).

Na página seguinte encontra-se uma tabela retirada da parte 2 do *standard* ISA-95 que referem a estrutura de diferentes classes que foram adotadas no modelo de dados dos DTO.

Tabela 5 - Descrição de estruturas de dados ISA-95

Nome dos atributos	Descrição	Exemplo
Classe *Personnel*		
Classe *Personnel*	Identifica a classe *Personnel* associada	*Widget Assembly Machine Operator*
Person	Identifica a classe *Person* associada	*SSN 999-55-1212*
Descrição	Contem informação adicional e a descrição do objeto	*"Widget machine operator availability over the 2000 New Year boundary"*
Capability Type	Um enumerado que pode conter os valores "*Available*", "*Unattainable*", "*Commited*"	*Available*
Reason	Define a razão para a *Capability Type*	*Available for Production*
Location	A identificação do modelo hierárquico associado ao elemento em questão. Campo opcional	*South Shore Production Plant*
Start Time	O tempo inicial associado à classe *Personnel*. Se omitido, será associado o *Start Time* da classe pai	*1999-12-30 11:59*
End Time	O tempo final associado à classe *Personnel*. Se omitido, será associado o *End Time* da classe pai	*2000-01-01 12:00*
Quantity	Define a quantidade associada à classe *Personnel*, se aplicável	*48*

Classe *Equipment*		
Equipment class	Identifica a classe *Equipment* associada	*Widget Jig*
Equipment	Identifica o equipamento associado	*Reactor 101*
Description	Contem informação adicional e descrições acerca da classe *Equipment*	*"Widget Jig commitment over the 2000 New Year boundary"*
Capability Type	Um enumerado que pode conter os valores "*Available*", "*Unattainable*" ou "*Commited*"	*Available*
Reason	Define a razão para a *Capability Type*	*Available for Production*
Location	A identificação do modelo hierárquico associado ao elemento em questão. Campo opcional	*South Shore Production Plant*
Start Time	O tempo inicial associado à classe *Personnel*. Se omitido, será associado o *Start Time* da classe pai	*1999-12-30 11:59*
End Time	O tempo final associado à classe *Personnel*. Se omitido, será associado o *End Time* da classe pai	*2000-01-01 12:00*
Quantity	Define a quantidade associada à classe *Personnel*, se aplicável	*48*

Classe *Process*		
ID	Um identificador único do processo	*1000104*
Description	Contem informação adicional e descrições acerca do processo	*"Widget Jig commitment over the 2000 New Year boundary"*
Capability Type	Um enumerado que pode conter os valores "*Available*", "*Unattainable*" ou "*Commited*"	*Available*
Reason	Define a razão para a *Capability Type*	*Available for Production*
Location	A identificação do modelo hierárquico associado ao elemento em questão. Campo opcional	*South Shore Production Plant*
Start Time	O tempo inicial associado à classe *Process*. Se omitido, será associado o *Start Time* da classe pai	*1999-12-30 11:59*
End Time	O tempo final associado à classe *Process*. Se omitido, será associado o *End Ti*me da classe pai	*2000-01-01 12:00*

Como se pode observar na Tabela 5, estão incluídos alguns dos modelos de metadados referentes ao *standard* ISA-95. De referir que alguns dos campos tiveram de ser adaptados e nem todos foram utilizados devido a facto de não fazerem sentido no contexto desta proposta de solução.

Com base no âmbito no processo de negócio da realização das ordens de fabrico, também foi desenvolvida uma implementação de um *Data Transfer Object* (DTO), com base nas nomenclaturas e estruturas de metadados do *standard* ISA-95 como se pode observar abaixo, na Figura 7.

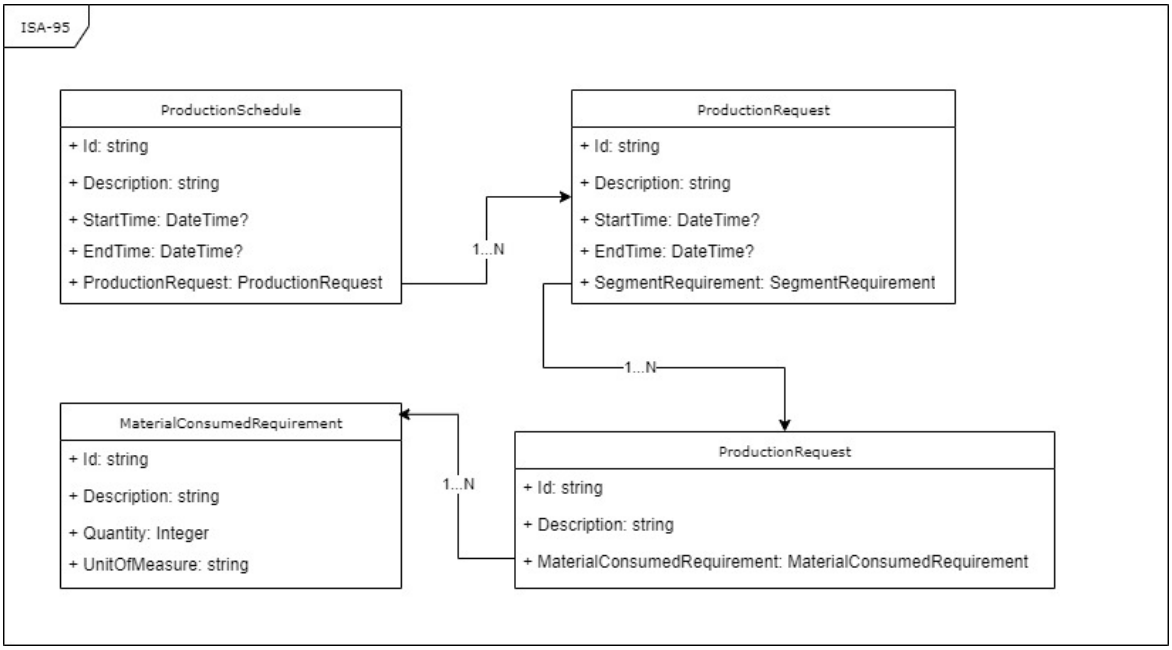

Figura 7 - Diagrama de classes referente ao DTO de uma ordem de fabrico.

Uma das informações mais requisitadas do ERP são as ordens de fabrico, e sendo esta informação necessária para o funcionamento das aplicações cliente do chão de fábrica, foi realizada a implementação da sua representação em formato ISA-95 (como se pode observar na Figura 7) e a implementação do esquema de classes em todos os projetos da solução Tooling 4G de forma a permitir a comunicação inter-serviço das ordens de fabrico.

Modelo de dados de persistência

Além do modelo de dados definido para os DTO, definiu-se adicionalmente o modelo de dados de persistência a ser utilizado no projeto *Intelligent Molds*, a ser persistido na base de dados como se pode observar na Figura 8, na página seguinte.

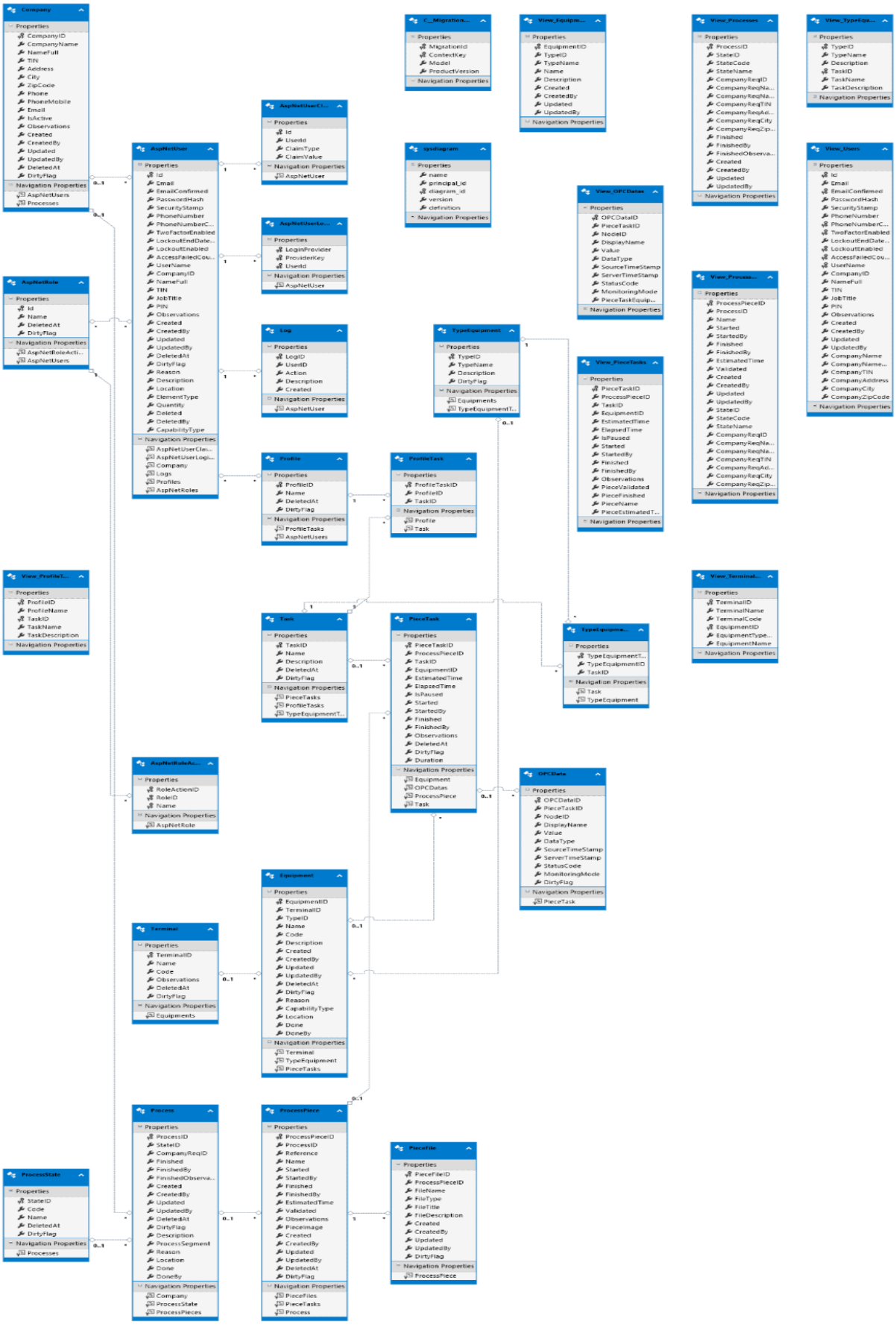

Figura 8 - Modelo de dados de persistência

4. Desenvolvimento da solução

Findado o levantamento de requisitos do qual originou o modelo de dados e o desenho da arquitetura da solução, procede-se o desenvolvimento da mesma. Neste capítulo é discutido cada um dos componentes que foram apresentados no capítulo da arquitetura da solução (capítulo 3.2) com um foco mais detalhado numa ótica técnica e de desenvolvimento.

4.1. WS_ERP

Um dos requisitos do projeto (RNF1 e RNF2) passava pela solução possuir compatibilidade com a maioria dos ERP do mercado. Para atingir um tal nível de compatibilidade, foi desenvolvido um serviço web WS_ERP cujo diagrama arquitetural está presente abaixo, na Figura 9.

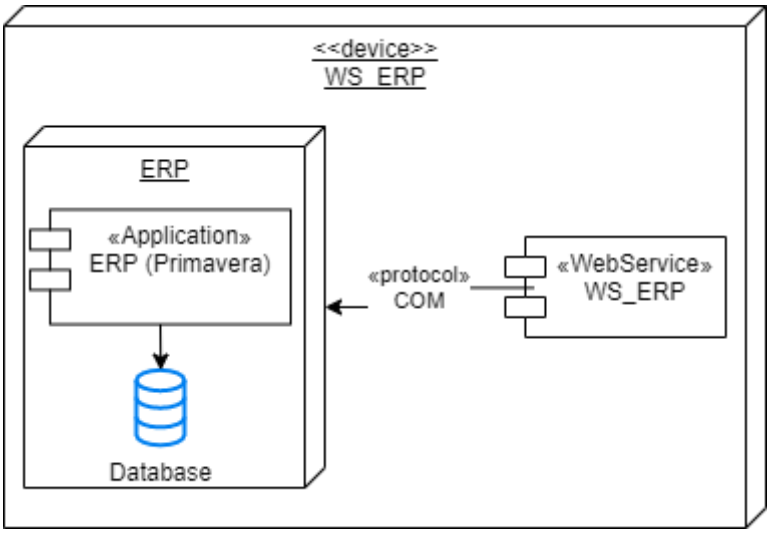

Figura 9 - Diagrama arquitetural do WS_ERP

O objetivo deste serviço é o de servir como "*driver*" para qualquer ERP, sendo que uma diferente implementação do mesmo existirá para uma diferente versão/*vendor* de um ERP. Em síntese, servindo como um *driver* ou mesmo uma *Remote Facade* de um ERP, este serviço disponibiliza como *interface* métodos *Create, Read, Update, Delete* (CRUD) sobre informação proveniente dos ERP.

Em termos arquiteturais, este serviço comporta-se essencialmente como uma *Application Programming Interface (*API), disponibilizando métodos para as suas operações CRUD sobre uma interface tecnológica REST.

Utilizou-se o padrão de desenho *Model-View-Controller* (MVC), em que as vistas correspondem às *interfaces* gráficas de comunicação com o utilizador (não aplicável neste contexto, pois este serviço não possui uma "vista"), os controladores correspondem às classes que recebem os pedidos API REST da aplicação, e o modelo que corresponde às principais entidades de domínio (nomeadamente as classes *Funcionário*, *Ordem Fabrico* e os repositórios onde os dados dos objetos destas classes são persistidos). Foi utilizada a *framework* .NET Core com a linguagem C#, sendo que tanto a linguagem como a *framework* foram escolha da *inCentea* – uma das empresas copromotoras do projeto.

Sendo que os dados de *input* necessários para este serviço são as ordens de fabrico e os dados dos funcionários (número mecanográfico e nome), este serviço teria de os adquirir num repositório onde os dados se encontram persistidos, que neste caso corresponde à base de dados de suporte a um ERP/MES. Sendo que para um dado ERP existem algumas formas diferentes de comunicação – via utilização de uma DLL, via acesso direto à sua base de dados (que foi utilizada neste estudo de caso) ou via Internet (*cloud)* – cada implementação é distinta tendo em conta a versão de um ERP/MES ou ERPs diferentes. Portanto, para se conseguir manter a compatibilidade deste serviço com a grande heterogeneidade de sistemas deste tipo onde se necessita adquirir os dados de *input*, terão de existir várias implementações, funcionando este serviço como um *driver* ou mais especificamente como uma *remote facade* [27].

4.2. WS_Tooling

Numa indústria onde a digitalização do chão de fábrica já se começa a tornar uma realidade do dia a dia, é cada vez mais comum encontrar cenários com soluções que envolvem sistemas de informação de gestão que permitem gerir *stocks*, processos de produção, entre outros.

Existindo uma crescente necessidade de digitalização e sendo estes sistemas peças fulcrais para suprir essa necessidade, é normal que tal procura tenha criado uma grande variedade de soluções no mercado. Cada solução com a sua implementação individual e o seu próprio modelo de domínio.

Sendo que o serviço WS_ERP (descrito no capítulo anterior – 4.1) é uma solução que permite o *fetching* de dados a partir de uma fonte de dados (i.e. ERP ou a base de dados deste), o tratamento desses mesmos dados não é uma função desse serviço, e sem referir que o modelo de domínio pode ter imensas inconsistências – diferentes fontes de dados podem ter modelos de domínio completamente diferentes – o que tornaria o processo de *parsing* – processamento dos dados – um processo extremamente penoso e pouco eficiente.

De forma a suprir a necessidade de centralizar o processo de *parsing* para um modelo de domínio "comum", e cumprindo com um princípio bastante comum na Engenharia de Software que implica que cada entidade (podendo esta ser uma classe, uma função ou mesmo um serviço) deve ter uma e só uma responsabilidade, optou-se então por criar um serviço web adicional – o WS_TOOLING, cujo diagrama arquitetural pode ser observado na Figura 10.

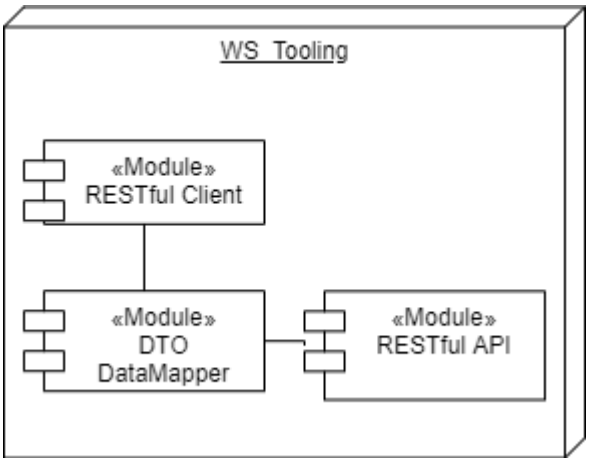

Figura 10 - Diagrama arquitetural do WS_TOOLING

Este serviço pode ser concetualmente dividido em três componentes – um *RESTful client*, um *Data Transfer Object* (DTO) *Data Mapper* e uma RESTful API.

4.2.1. RESTful Client

No exemplo de algum componente do chão de fábrica tentar obter informações relativas às ordens de fabrico, o serviço WS_Tooling vai emitir um pedido para a API do WS_ERP por forma a obter os dados em estruturados como um DTO. Estes dados vêm normalmente em formato *JavaScript Object Notation* (JSON) ou *eXensible Markup Language* (XML).

Ao receber os dados, que estão estruturados no formato do próprio ERP, torna-se necessário normalizar esse formato para que as aplicações clientes que os utilizarão para representação de gráficos de produtividade ou para receção das ordens de fabrico não tenham de ter um *parser* para cada ERP existente no mercado. É neste sentido que o WS_Tooling, ao receber os dados, fará a conversão da estrutura dos mesmos para um formato *standard* na indústria (ISA-95).

4.2.2. DTO Data Mapper

Sendo a transformação e estruturação de dados algo complexo e moroso, várias propostas foram colocadas sobre análise de forma a escolher a que solucionasse o problema de forma mais rápida e eficiente.

Sendo os padrões de desenho da engenharia de software modelos concetuais altamente testados e cuja sua eficácia está comprovada em incontáveis ambientes de produção, esta pareceu ser a solução óbvia e depois de alguma análise, dois padrões de desenho pareceram interessantes à solução deste problema – *adapter pattern* (Figura 11) e *data mapper* (Figura 12).

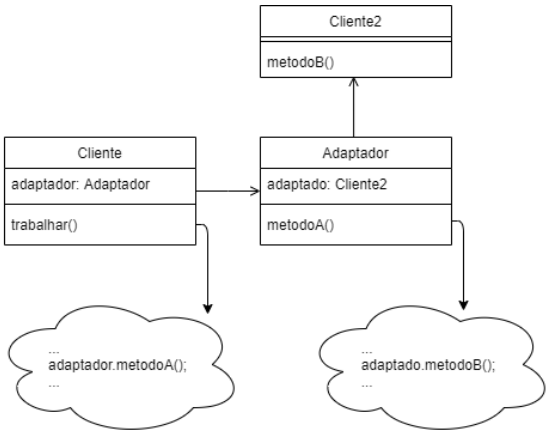

Figura 11 - Exemplo do padrão de desenho *adapter pattern*

O objetivo do *adapter pattern*, ou padrão adaptador em português, é a ligação de duas entidades (ou classes em termos utilizados no paradigma de programação orientada a objetos – POO) que são estruturalmente incompatíveis.

Por analogia, seria como tentar ligar dois cabos que não são compatíveis, sendo que para o efeito se utiliza um adaptador. Neste caso em concreto, o objetivo é transformar um tipo de dados proveniente de um ERP para um tipo de dados ISA-95.

Um dos problemas desta abordagem é a grande quantidade de código necessária para atingir o fim da transformação dos dados, sendo que para cada ERP que se pretende que seja compatível com esta solução, terá de existir uma *interface* e uma classe que teriam de ser implementados, juntamente com todo o código para o processo de *parse*. Esta solução foi inicialmente tentada, mas demonstrou-se que seria bastante extensa e sendo que um dos objetivos da solução era o de uma fácil manutenção, esta proposta de solução não seria aparentemente a mais eficiente. Dando origem a uma solução baseado no padrão de desenho *data mapper* (Figura 12)

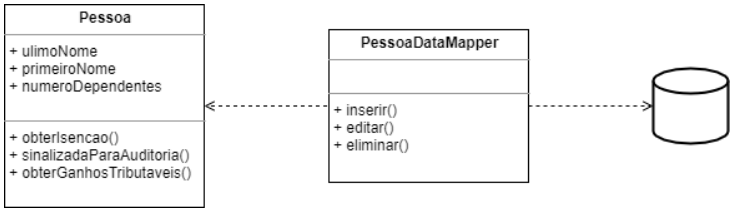

Figura 12 - Exemplo do padrão de desenho Data Mapper

O objetivo do padrão de desenho *data mapper* é o mapeamento dos dados que estão em memória para um sistema de persistência como uma base de dados. Esta solução existe principalmente devido às inconsistências concetuais na forma como os dados são representados em memória ou numa base de dados, sendo que por norma os dados em memória são representados por objetos e na grande maioria das bases de dados num formato relacional – de forma a poder representar-se não só os dados como as suas relações. Quando se pretende persistir estes dados, normalmente é necessário existir uma camada arquitetural que faça a transformação estrutural desses mesmos dados mantendo a sua coerência e sincronização entre a memória do servidor e uma fonte de dados persistentes.

Tipicamente para se resolver este problema, criam-se módulos que normalmente se encontram em qualquer *framework* das linguagens de programação mais utilizadas do mercado, denominadas de *Object-Relational Mapping* (ORM). Uma das estratégias que as ORM utilizam para resolver este problema de incoerência é utilizar padrões de desenho, sendo que um deles é precisamente o *data mapper*.

Outra das utilizações de um *data mapper* é este poder ser utilizado como um "mapeador" entre dois formatos diferentes, que é exatamente o que se pretende neste caso. Uma das vantagens também de utilizar um *mapper* é que existem inúmeras implementações deste padrão de desenho em praticamente todas as linguagens de programação mais utilizadas do mercado.

Como tal, fez-se um estudo e decidiu-se que a implementação de *data mapper* em C# a utilizar-se no projeto seria a *AutoMapper* [28]. Com esta versátil biblioteca, não é necessário a criação de uma classe por cada ERP (ou versão de ERP), sendo que só é necessário criar uma classe que irá conter todas as regras que o *AutoMapper* utilizará para fazer o *parse* de uma classe proveniente do ERP para uma classe que empregue o *standard* ISA-95.

4.2.3. RESTful API

De forma a que todos os serviços que tenham uma dependência dos dados provenientes do ERP não tenham de realizar pedidos ao serviço *WS_ERP* e depois realizar o *parsing* desses mesmos dados, o serviço *WS_Tooling* expõe uma API REST para que se consiga facilmente aceder aos dados, devolvendo os mesmos num formato *standard* ISA-95 em formato JSON.

O *standard* ISA-95 é algo extenso, dividido em 5 partes:

- ***ANSI/ISA-95.00.01-2000, Enterprise-Control System Integration Part 1: Models and Terminology*** **–** Define a terminologia *standard* e define qual a informação a ser transmitida;

- ***ANSI/ISA-95.00.02-2001, Enterprise-Control System Integration Part 2: Object Model Attributes*** **–** Define todos os atributos referentes a todos os objetos;

- ***ANSI/ISA-95.00.03-2005, Enterprise-Control System Integration, Part 3: Models of Manufacturing Operations Management*** **–** Foca nas funções e atividades ao nível do processo de produção e dos *Manufacturing Execution Systems* (MES);

- ***ISA-95.04 Object Models & Attributes Part 4 of ISA-95: "Object models and attributes for Manufacturing Operations Management"*** **–** Define quais dados devem ser transmitidos entre sistemas MES;

- ***ISA-95.05 B2M Transactions Part 5 of ISA-95: "Business to manufacturing transactions"*** **–** Define qual informação deve ser transmitida entre os equipamentos de gestão e os de produção.

No âmbito deste projeto, houve um maior foco na parte 2 deste *standard*, de forma a definir quais os objetos e atributos a constarem nos *Data Transfer Objects* (DTO), que são os objetos a serem serializados e transferidos entre os diferentes serviços existentes no projeto.

4.3. OEE Web Server

Numa fábrica de moldes, existem diferentes fatores que podem influenciar a eficiência do processo de produção, incluindo a eficiência das máquinas envolvidas nas linhas de produção ou a produtividade laboral dos colaboradores.

Uma maior eficiência no processo de produção traduz-se num maior número de peças produzidas numa unidade de tempo, bem como na minimização da produção de peças defeituosas. Como tal, muitas empresas na área industrial criaram sistemas versáteis de análise da qualidade dos produtos produzidos bem como sistemas de predição de falha dos equipamentos, de forma a tentarem manter ao máximo a eficiência das suas linhas de produção.

4.3.1. Indicadores de Desempenho

Tendo, portanto, a produtividade dos equipamentos que poderão estar num chão de fábrica como uma variável a otimizar, surgiu a necessidade de se utilizarem indicadores de desempenho, conhecidos na literatura por *Key Performance Indicators* (KPI). A utilização de KPI torna-se interessante pelo facto de exigir poucas mudanças no ambiente de produção – este modelo é puramente matemático – ao contrário de, por exemplo, a utilização de sistemas de predição de falhas, que estão ativamente a analisar o ambiente de produção de forma a detetar padrões de trabalho irregulares para poder relatar uma falha eminente.

Se considerarmos os KPI como um conjunto de indicadores que preveem quantificar o desempenho de uma planta industrial, podemos pensar num subconjunto de KPI em específico que é muito indicado para ambientes industriais – o *Overall Equipment Effectiveness* (OEE). Esta relação é ilustrada com um diagrama de *Venn* na Figura 13.

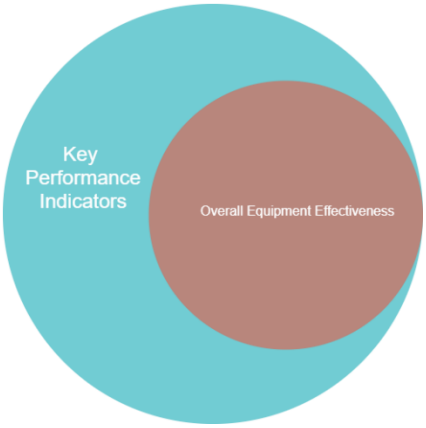

Figura 13 - Diagrama de Venn da relação KPI - OEE

Para reduzir paragens de máquinas, gastos desnecessários e aumentar a produtividade, em 1988, *Nakajima* [29] criou um sistema revolucionador chamado *Total Productive Maintenance* (TPM) cujo objetivo era o de aumentar o desempenho do processo de produção. Nesse sistema constava uma métrica quantitativa a qual ele chamou de *Overall Equipment Effectiveness* (OEE).

Esta mede, em percentagem, a quantidade efetiva de eficiência produtiva. O OEE foca-se em três princípios para a medição da qualidade: disponibilidade, desempenho e taxa de qualidade.

A aplicabilidade do OEE pode variar dependendo do tipo de indústria que o está a utilizar. Desta forma, foram criadas algumas variantes do OEE original (e.g. *Overall Factory Effectiveness* (OFE), *Overall Plant Effectiveness* (OPE), *Production Equipment Effectiveness* (PEE), etc.) de forma a conseguir-se enquadrar o conceito nas necessidades das respetivas indústrias [30].

Sendo a utilização de OEE um requisito bem definido nos objetivos deste projeto, para se conseguir realizar uma efetiva previsão de produtividade de uma indústria, foi idealizado um módulo que visa a monitorização de indicadores de produtividade (fornecidos por outro parceiro do projeto – o CENTIMFE) e nos quais os OEE estão incluídos.

Os OEE podem ser calculados a partir da seguinte fórmula (em percentagem):

1. $Overall\ Equipment\ Effectiveness\ (OEE) = DO\ x\ P\ x\ Q$

Em que DO significa disponibilidade, P é a taxa de desempenho e Q a taxa de qualidade.

- **Disponibilidade** – Tem em conta todos os eventos que contribuem para um atraso da produção. Obtém-se a partir do rácio *Tempo Bruto de Funcionamento (TBF) / Tempo Funcionamento Necessário (TFN)* e é expresso em percentagem;
- **Taxa de desempenho** – Tem em conta eventos que possam causar uma diminuição do desempenho do processo de produção. Obtém-se a partir do rácio *Tempo Líquido de Funcionamento (TLF) / Tempo Bruto de Funcionamento (TBF)* e é expresso em percentagem;
- **Qualidade -** Tem em conta peças que foram construídas, mas que não vão de encontro aos *standards* da empresa, bem como as que necessitam de ser retrabalhadas. Obtém-se a partir do rácio *Taxa Líquida de Funcionamento (TLF) / Tempo Útil (TU)* e é expresso em percentagem.

Em adição aos OEE, o CENTIMFE também requisitou para este *dashboard* a *Taxa de Rendimento Global (TRG)* que podem ser calculados a partir da seguinte fórmula (em percentagem):

2. *Taxa de Rendimento Global (TRG) = Tempo Útil (TU) / Tempo Trabalho Planeado (TTP)*

Em que o *Tempo Útil (TU)* corresponde ao tempo de transformação necessário para o fabrico de peças conformes e o *Tempo Trabalho Planeado (TTP)* corresponde à amplitude de tempo de trabalho que o equipamento pode operar.

4.3.2. Descrição das funcionalidades

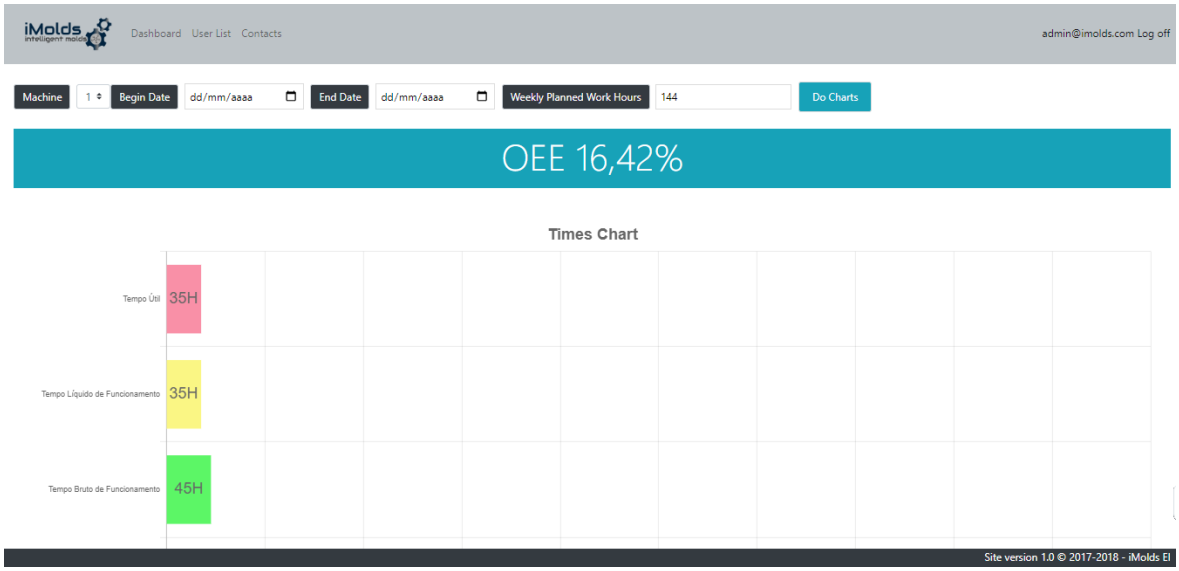

Figura 14 - Gráfico demonstrador de OEE

Como se pode observar na Figura 14, o *dashboard* permite a visualização dos OEE e respetivos indicadores constituintes numa série temporal (na imagem observa-se o *Tempo Útil (TU)*, o *Tempo Líquido de Funcionamento (TLF)* e o *Tempo Bruto de Funcionamento (TBF)*). Sendo que os dados exibidos na figura, têm um efeito puramente demonstrativo, sendo associados a um processo de produção de teste.

É possível neste protótipo funcional, a criação e visualização de gráficos, optando pelo número da máquina e também por uma filtragem por datas a partir dos campos "*Begin Date*", "*End Date*" e "*Weekly Planned Work Hours"*. Para gerar o gráfico, basta clicar no botão "*Do Charts*".

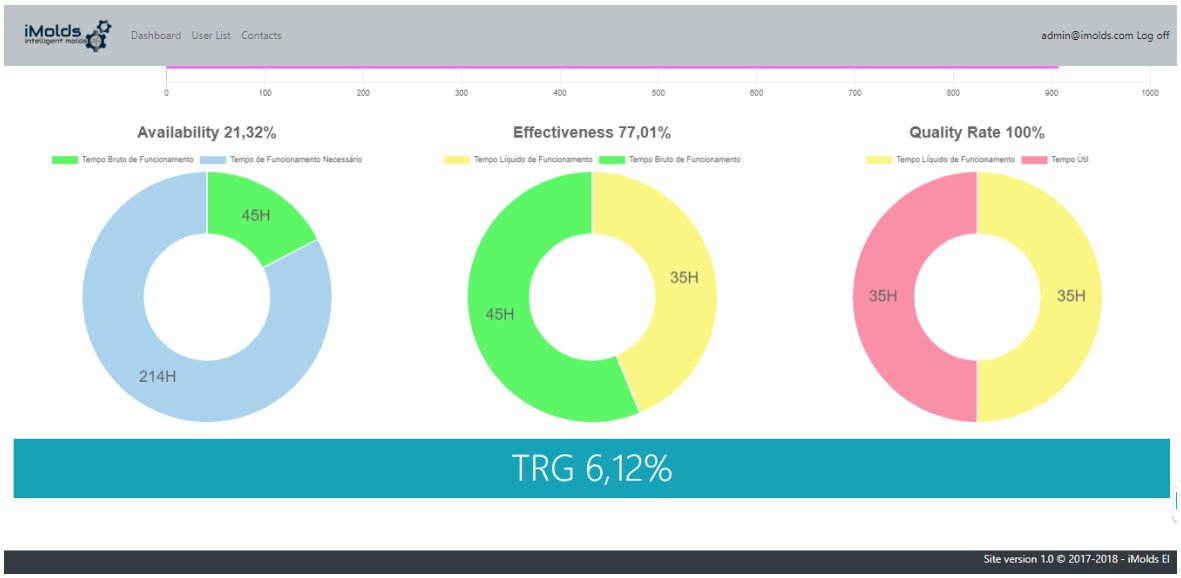

Figura 15 - Gráfico demonstrados com alguns OEE

Na Figura 15, é possível observar gráficos referentes à disponibilidade (*availability*), taxa de desempenho (*effectiveness*) e qualidade (*quality rate*), bem como a TRG (*Taxa de Rendimento Global*) a azul, para determinado equipamento/máquina. Estes gráficos também foram gerados a partir da filtragem do número da máquina e de um período temporal (que é um filtro opcional).

A partir deste protótipo funcional desenvolvido, é possível de forma simples a criação de gráficos com informação em tempo real de forma a que se possa detetar qualquer anomalia numa linha de produção, seja por exemplo por uma máquina estar em eminência de falha, seja para garantir que todo o sistema de produção mantém uma velocidade e eficiência constante de qualidade de produção ou assemblagem.

Este protótipo funcional permite também o acesso à lista de utilizadores que estão registados e que podem aceder ao *dashboard*. Também é possível a realização das operações *Create*, *Read*, *Update*, *Delete* (CRUD) sobre os utilizadores, como se pode observar na Figura 16.

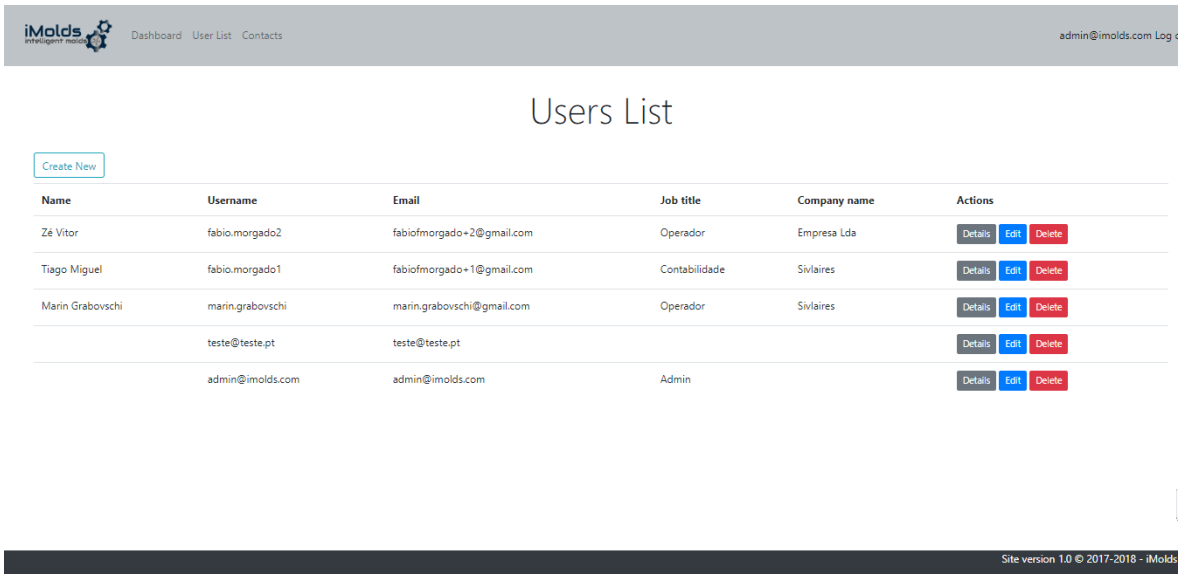

Figura 16 - Lista de utilizadores

Esta lista pode ser acedida clicando no *link "User List"* através do menu superior, junto ao ícone no canto superior esquerdo da página. O utilizador poderá visualizar a lista completa de todos os utilizadores do *dashboard*, tendo também à sua disposição as operações CRUD acima referidas, sendo que para criar um utilizador novo, basta clicar no botão "*Create New*" do lado superior esquerdo, em cima da lista de utilizadores. Nesta página, será solicitado ao utilizador a inserção de alguns dados acerca do utilizador como por exemplo o seu nome, *username*, e-mail, entre outros.

De referir que estas ações referidas só são possíveis porque se acedeu ao *dashboard* com uma conta de administrador. Se fosse uma conta de utilizador normal, o acesso aos dados de outros utilizadores seria restrito.

4.3.3. Detalhes técnicos

De um ponto de vista técnico e de implementação, este protótipo funcional OEE *Web Server*, que pertence a toda a arquitetura do projeto *Tooling 4G*, consiste numa solução ASP.NET MVC.

O motivo pelo qual recaiu a escolha nesta tecnologia, prende-se com a maturidade desta mesma *framework* na data em que se iniciou o desenvolvimento desta solução (2018), pela utilização da linguagem C# que permite que esta aplicação seja mais facilmente mantida pela equipa de desenvolvimento do parceiro tomador da tecnologia do projeto, a inCentea (que utiliza maioritariamente desenvolvimento em C#) e por fim a utilização do padrão arquitetural *Model-View-Controller* (MVC), que permite uma maior divisão de responsabilidades entre as entidades (classes) do código fonte, bem como assegura uma maior manutenção do código.

O padrão arquitetural MVC é um padrão arquitetural de lógica de apresentação que age ao nível estrutural do código fonte de um programa (i.e., na estrutura em como o código é organizado). Uma arquitetura MVC permite uma divisão mais coesa do código, evitando a duplicação do mesmo, facilitando a manutenção com alterações que são específicas ao que se quer alterar (e.g. para alterar o *design* de um botão, só é necessário mexer na classe associada à "vista" do mesmo, e não à classe "controlador" que está associada ao seu comportamento). Pelas razões apresentadas, a escolha de uma *framework* MVC em C# para esta solução foi a decisão predileta.

Para a criação de gráficos, foi utilizada a biblioteca *Highcharts* para .NET [31]. Sendo que esta biblioteca não tinha uma licença que permitia a sua utilização de forma gratuita, mais tarde migrou-se para a biblioteca *Charts.js* [32].

Os gráficos são gerados com base no cálculo dos OEE, sendo que para a aplicação poder demonstrar os gráficos no ecrã, tem de ter acesso à informação relativa à produção da fábrica como tempos de máquinas, entre outros.

Estes tempos são adquiridos a partir da rede OPC-UA, sendo posteriormente guardados numa base de dados associada a esta aplicação. Para serem visualizados em formato gráfico, os dados são processados por um controlador (no contexto do padrão MVC), que fará as transformações necessárias a esses dados, como a redução da granularidade destes de forma a que se possa utilizá-los nas fórmulas que irão originar os OEE. Tendo o controlador realizado todo o processamento dos dados, estes são apresentados pela vista que utilizará a biblioteca *Charts.js* para a sua exibição.

Em relação à gestão de funcionários, a origem dos dados destes existe tipicamente na base de dados do protótipo funcional *OEE Web Server*. Numa primeira instância, os dados terão de ser carregados manualmente para a base de dados com a utilização de *seeders* ou *scripts* de carregamento. Mas, existindo uma conta administrativa, é possível através da aplicação, como se pode observar na Figura 16, a inserção, edição, eliminação e listagem de todos os funcionários.

Num contexto mais técnico, os dados dos funcionários encontram-se ou serão armazenados numa base de dados *SQL Server* 2017. O acesso a estes dados será feito pelo *Object-Relational Mapping* (ORM) *Entity Framework* (EF) [33], que é mantido pela Microsoft. Um ORM é um componente de software que permite o mapeamento entre uma base de dados relacional e um programa que utilize o paradigma de programação orientada a objetos (POO). Utilizando a EF, a cada tabela da base de dados corresponde uma classe. Dado como um exemplo uma tabela "Pessoas", iriamos ter uma classe "Pessoa" correspondente, associada ao modelo da nossa aplicação dentro do contexto MVC. Independentemente do modelo, existe um controlador que disponibilizará os serviços que possibilitam as operações *Create-Read-Update-Delete* (CRUD). Para a vista, não foi utilizada nenhuma *framework* adicional.

Como trabalho futuro, era interessante que a origem dos dados fosse exclusivamente o ERP, servindo a base de dados local do *OEE Web Server* como uma *cache* que permitiria o funcionamento da aplicação temporariamente sem acesso ao ERP (i.e., modo *offline*). Mas tal solução também implicaria uma complexidade adicional devido à implementação da lógica de *cache* no sistema. A utilização de uma cache ao nível aplicacional para este cenário, implicaria um levantamento das possíveis aplicações que pudessem aceder e modificar os dados dos funcionários que o protótipo funcional *OEE Web Server* iria consumir, bem como a periodicidade do consumo dessa informação.

Com base no resultado desse hipotético estudo, teriam de ser estudadas várias soluções que se adequassem a esse problema de forma a conseguir evitar o máximo de *cache misses* – quando os dados que a aplicação pretende aceder não se encontram em cache e é necessário ir obtê-los diretamente a partir da fonte de dados.

4.4. Rede OPC-UA

Uma das premissas que a Indústria 4.0 vem trazer às fábricas, consiste não só na digitalização do chão de fábrica, mas também numa revolução na forma como as linhas de produção trabalham. Sendo que o objetivo nestes cenários é diminuir o máximo de interação humana possível, com o intuito de reduzir o erro humano e também de acelerar o processo de produção (o *input* humano é, tipicamente, lento e sujeito a erros). Consequentemente é expetável tornar a passagem de informações entre linhas de produção, entre postos de fabrico, entre máquinas e até mesmo entre sensores o mais automatizada possível, sendo que no limite, seria interessante e expetável que as próprias máquinas tivessem inteligência o suficiente para elas próprias transferirem a informação que acham correta, com uma granularidade aceitável – não transferir os dados por exemplo da posição de uma *Computer Numeric Control* (CNC) cinquenta vezes por segundo, sendo que nenhuma outra máquina ou *web service* da fábrica necessita dessa informação atualizada com tanta frequência.

Sendo a comunicação máquina-máquina ou *Machine to Machine* (M2M) como é conhecida na literatura o passo seguinte, foi realizado em âmbito de revisão da literatura, um estudo sobre os protocolos informáticos para transferência de dados em ambiente industrial, sendo que o protocolo que melhor se enquadra para realizar este tipo de comunicação que é por natureza assíncrona, é o protocolo *Open Platform Communications - Unified Architecture* (OPC-UA) [36].

O protocolo OPC-UA teve a sua primeira versão a público em 2008 [36], tendo sido uma evolução do protocolo já descontinuado OPC [3], que tinha imensas variantes e sendo este novo protocolo o resultado de um esforço para unificar todas essas variantes num único protocolo (*i.e.* OPC-UA). Criado e regulado pela *OPC Foundation*, foi definido como um protocolo baseado na arquitetura *Service Oriented Architecture* (SOA), M2M e compatível com várias plataformas, sendo esta uma das grandes vantagens comparativamente com o seu ancestral (OPC) que só suportava as plataformas *Windows.*

Também é definido como um protocolo seguro, possuindo várias camadas de segurança e obedecendo ao modelo *CIA TRIAD* (ver **Erro! A origem da referência não foi encontrada.**).

4.4.1. Detalhes do protocolo OPC-UA

Num ambiente fabril, existem sistemas de informação que se inserem mais facilmente em certos processos de negócio. Se se idealizar os sistemas utilizados num ambiente fabril de forma hierárquica, iria-se obter por exemplo o ERP no topo da pirâmide e os sistemas de aquisição de dados de chão de fábrica na base dessa mesma pirâmide. O OPC-UA neste âmbito, pode interagir desde o nível mais baixo (chão de fábrica) até ao nível mais alto (ERP) como se pode observar na Figura 17.

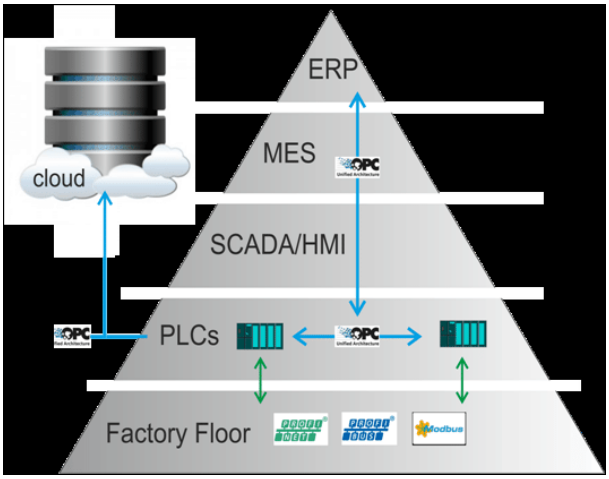

Figura 17 - Pirâmide de níveis de tecnologia de uma fábrica (adaptado de [37]**)**

Na Figura 17 pode observar-se que os dados são transferidos dos sistemas de informação do chão de fábrica (tipicamente de sensores, máquinas, bancadas e/ou linhas de produção) para os *Programming Logic Units* (PLC) que gerem essas mesmas máquinas. Os protocolos que gerem esta comunicação máquina – PLC são normalmente proprietários. Mas por vezes existe a necessidade de PLC comunicarem entre si, seja por exemplo por se ter definido um alarme num PLC e o outro necessitar de saber quando a tarefa está completa para iniciar a sua própria tarefa. Este tipo de comunicação é possível e até aconselhada de se utilizar pelo protocolo OPC-UA como é representado pela Figura 17.

Os dados gerados pelos PLC também podem ser enviados para um servidor na *cloud* com o objetivo de serem guardados num ou mais *data warehouses* ou armazéns de dados (um sistema de informação cujo intuito é o de armazenar grandes volumes de informação), para mais tarde poder ser utilizado em por exemplo análises para se extrair conhecimento a partir da informação e permitir assim aos decisores tomarem as decisões mais corretas nos seus planos estratégicos.

Sendo que muita da informação gerada é necessária no ERP, ou em um *Manufacturing Execution System* (MES), o OPC-UA também poderá ser utilizado para este propósito de forma simplificada.

O protocolo OPC-UA possui um *standard* dividido em cerca de 20 partes [38], sendo que cada parte se foca especificamente numa função do protocolo (e.g. endereçamento, tipo de dados, atributos dos dados transferidos, entre outros).

A *OPC Foundation* [39] define o protocolo OPC-UA como um protocolo multicamada, independente de plataforma, e com os seguintes objetivos:

Equivalência funcional

- Permite a descoberta de servidores OPC-UA que se encontrem disponíveis, encontrem-se esses servidores no computador local ou na rede local;
- Representação hierárquica de dados (e.g. ficheiros e pastas), permitindo realizar a composição de estruturas complexas a partir de estruturas mais simples e possibilitando facilmente a descoberta destes dados por servidores/clientes OPC-UA;
- Leitura e escrita de dados com base em permissões de segurança;
- Monitoramento de dados ou informação e criação de alertas no caso de algum dos dados apresentar valores fora do normal (definido por critérios do utilizador);
- Clientes OPC-UA podem executar programas baseados em métodos definidos no servidor.

Independência de plataforma

O protocolo OPC-UA foi construído tendo em conta a grande heterogeneidade de *hardware* no mercado e também de sistemas operativos, sendo que é compatível em servidores na *cloud*, PLC, microcontroladores (inclusive os que possuem arquitetura *Acorn RISC Machine* (ARM)), ou computadores embutidos entre outros dispositivos.

Segurança

No contexto de segurança, OPC-UA é um protocolo que possui uma grande variedade de mecanismos de segurança de forma a garantir um transporte extremamente seguro de dados:

- No contexto de transporte de dados, é possível o transporte utilizando o protocolo *OPC-binary transport*, ou até mesmo utilizando *JavaScript Object Notation* (JSON) sobre a tecnologia *websockets;*
- Criação de um canal de comunicação seguro e encriptado a vários níveis de encriptação;
- Assinatura da mensagem que permite garantir a integridade das mensagens e o princípio do não repúdio;
- Sequenciamento de pacotes, o que permite evitar ataques *replay;*
- Autenticação, que é realizada entre um servidor e cliente OPC-UA utilizando um certificado X509, o que também permite definir níveis de acesso de determinado cliente a determinada informação;
- Controlo de utilização, que permite a restrição a determinados *address spaces* ou espaços de endereçamento que tenha informação ou dados que o utilizador autenticado não deveria ter acesso;
- Auditoria, que permite o monitoramento dos acessos, bem como de todas as transações que são realizadas.

Extensibilidade

Sendo o OPC-UA um protocolo com uma arquitetura multicamada, é possível a inserção de novos algoritmos ou esquemas de segurança, bem como protocolos de transporte de dados ou até mesmo serviços aplicacionais como se pode observar no esquema na Figura 18.

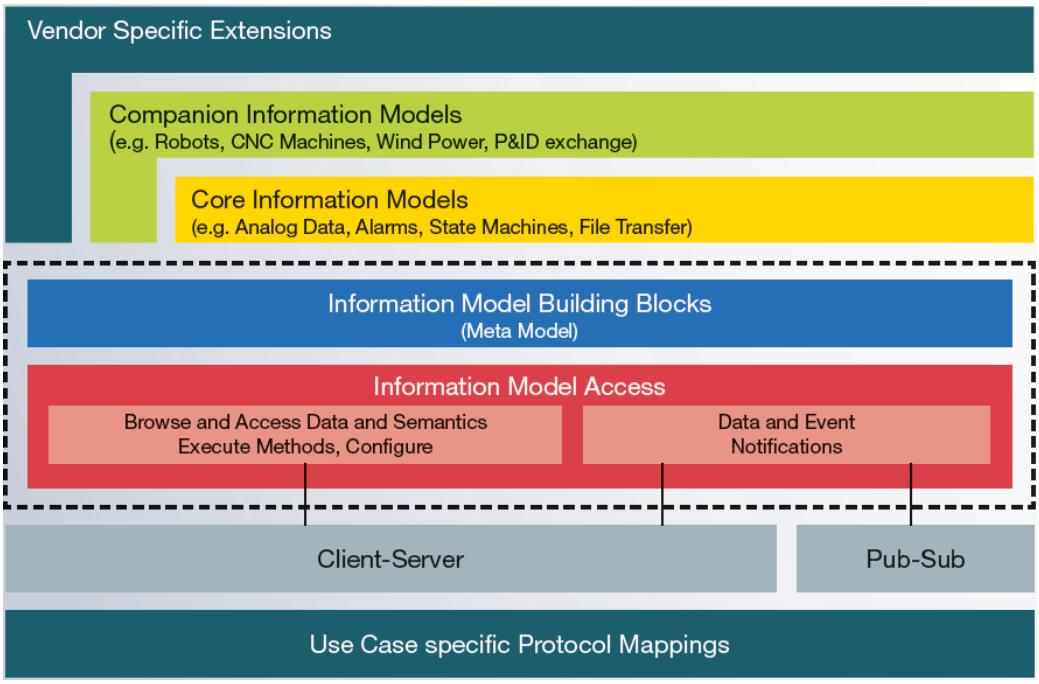

Figura 18 - Modelo de extensibilidade do protocolo OPC-UA (retirado de [36]**)**

A comunicação com o protocolo OPC-UA pode ser feita de duas formas – seja no modelo arquitetural cliente servidor, ou no modelo *publisher subscriber*. No primeiro modelo referido, é iniciado uma sessão entre os dois sendo que a informação trocada tem uma natureza síncrona, enquanto que com o último modelo, a informação trocada não recai numa sessão entre o cliente e o servidor, mas sim em mensagens trocadas assincronamente.

Criando uma sessão é possível aceder aos dados que se encontram no *address space* do servidor. Podendo ser possível também criar alarmes ou definir eventos. No modelo *Pub-Sub*, é possível a notificação de eventos ou dados e a sua subscrição.

4.4.2. Equipamentos sem OPC-UA

A utilização do protocolo OPC-UA implica a existência de pelo menos um servidor e um cliente OPC-UA. Tal cenário é simplificado quando as máquinas que funcionam ao nível do chão de fábrica possuem compatibilidade de fábrica com o protocolo OPC-UA, mas quando esta realidade não acontece – o que é efetivamente comum, sendo que o protocolo é relativamente novo comparado com outros protocolos que já são utilizados há décadas em âmbito industrial – é necessária uma alternativa que permita a transferência de dados de um formato *ad-hoc* para OPC-UA.

Um dos casos onde este problema se evidenciou, foi no caso onde se desenvolveu uma aplicação de recolha de dados referentes a ordens de fabrico para o chão de fábrica. Um dos requisitos desta aplicação era a aquisição de informação provenientes do ERP ou de um MES. Em contrapartida à programação de um canal de comunicação direto com o ERP, optou-se pela utilização do protocolo OPC-UA para esse efeito.

A solução passou pela criação de um servidor e de um cliente OPC-UA.

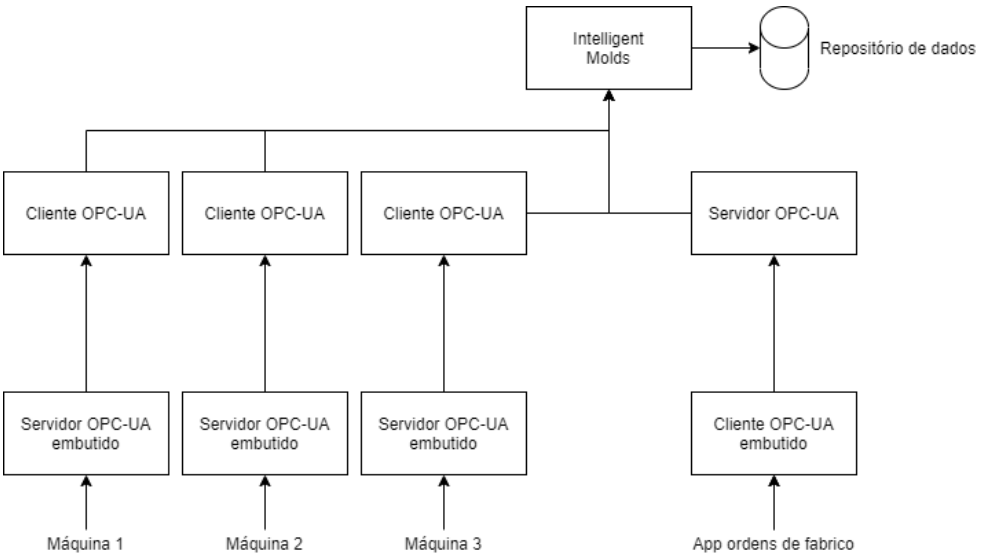

Figura 19 - Modelo arquitetural simplificado da comunicação OPC-UA com o serviço *Intelligent Molds*

A Figura 19, servindo de complemento para a Figura 5, pretende ilustrar de forma mais detalhada como vários componentes OPC-UA podem se ligar, via rede OPC-UA ao *Intelligent Molds* (*dashboard* OEE), ilustrando assim a hierarquia de comunicação entre as máquinas que operam ao nível de chão de fábrica, utilizando nativamente o protocolo OPC-UA, como servidores e a existência de clientes OPC-UA para o consumo desses mesmos dados – os clientes servem como intermediários entre as máquinas e a aplicação *Intelligent Molds.*

No caso da máquina ou o posto de trabalho não possuir compatibilidade nativa com o protocolo OPC-UA, como se pode observar na imagem (Figura 19), identificado como a *app* de ordens de fabrico, a direção da comunicação funciona no serviço inverso – a *app* irá pedir, via OPC-UA ao servidor, todas as ordens de fabrico mais recentes para as exibir ao utilizador. O servidor OPC-UA ao receber o pedido do cliente, irá consumir uma *Application Programming Interface* (API) que a solução *OEE Web Server* possui que por sua vez irá transferir o pedido de ordens de fabrico para o ERP. Desta forma, a solução *OEE Web server* funciona como uma "fachada" (i.e., padrão *facade*) e abstrai a existência do ERP neste cenário de comunicação. Esta abordagem é útil se futuramente se quiser que as ordens de fabrico provenham de outro local além do ERP (e.g. um MES por exemplo), sendo só necessário alterar os *endpoints* no *OEE Web Server* e não no servidor OPC-UA que serve a *app* de ordens de fabrico.

4.4.3. Armazenamento de dados provenientes da rede OPC

Como foi descrito no capítulo 4.3, a solução *OEE Web Server* – aplicação com *dashboard* dos OEE e painel administrativo dos utilizadores – pertencente à arquitetura *Tooling 4G*, realiza o cálculo dos indicadores de desempenho com base em fórmulas assentes em tempos realizados e planeados por máquinas. Sendo que cada máquina pode ser servidora dos seus próprios dados, metodologias arquiteturais habituais como a cliente-servidor não eram as mais indicadas para este cenário, logo e tirando partido da descentralização na comunicação do protocolo OPC-UA, pensou-se em enviar todos estes dados, assincronamente, para a base de dados da solução *OEE Web Server*, respeitando, para isto, o *standard* ISA-95 [40] que define como são estruturados os dados provenientes do OPC-UA e como estes serão armazenados no ERP.

4.5. Cliente Angular

Sendo a comunicação *Machine to Machine* (M2M) um dos pontos fulcrais deste projeto, e principalmente devido à necessidade de aquisição de dados sensoriais para o cálculo dos OEE, de forma a conseguir-se monitorizar o desempenho do chão de fábrica em tempo real, subsistia também a necessidade de todos os sensores, dispositivos e máquinas comunicarem utilizando o protocolo e *standards* que foram definidos para este projeto – o protocolo OPC-UA.

Um dos pontos positivos que pode suportar uma maior adoção futura do protocolo OPC-UA, prende-se com este já ser suportado nativamente por algumas máquinas que operam em chão de fábrica. Mas sendo a indústria uma das áreas de negócio que tem a maior resistência à mudança, é expetável que um cenário onde o OPC-UA seja um padrão plenamente aceite não se realize num futuro próximo.

Posto isto, e tendo em conta que este projeto serve o propósito de uma *framework* a ser utilizada em vários ambientes de chão de fábrica com níveis de digitalização variados, identifica-se a necessidade de sensores/dispositivos/máquinas comunicarem com a restante infraestrutura digital do chão de fábrica, utilizando o protocolo OPC-UA. Isto exige, por si só, uma solução que permita esta comunicação inter-protocolar.

Tendo em conta que numa primeira instância são os operadores que operam estas máquinas, a solução partiria por ser uma aplicação com uma *interface* com o utilizador que permitisse a inserção manual dos dados da ordem de trabalho a ser realizada. Desta forma seria possível a integração de equipamentos "não OPC-UA", de uma forma intuitiva do ponto de vista do utilizador desta solução, sem os custos de aquisição de novo material.

Numa empresa de moldes, uma ordem de trabalho ou ordem de fabrico consiste num comando que pode incluir a requisição para maquinar determinada peça ou para efetuar a montagem de um conjunto de peças de um molde, sendo que associado a este comando – que tipicamente pode ser um documento – podem estar as especificações do item, o seu código e descrição, quantidade a ser produzida e a data de entrega prevista.

Numa primeira instância, realizou-se um esboço informal do processo empresarial de modo a conseguir identificar todas as fases associadas a este processo. Este trabalho de identificação das fases do processo foi realizado em conjunto com a *inCentea* que é uma empresa com bastante experiência acumulada em soluções em produção desta natureza.

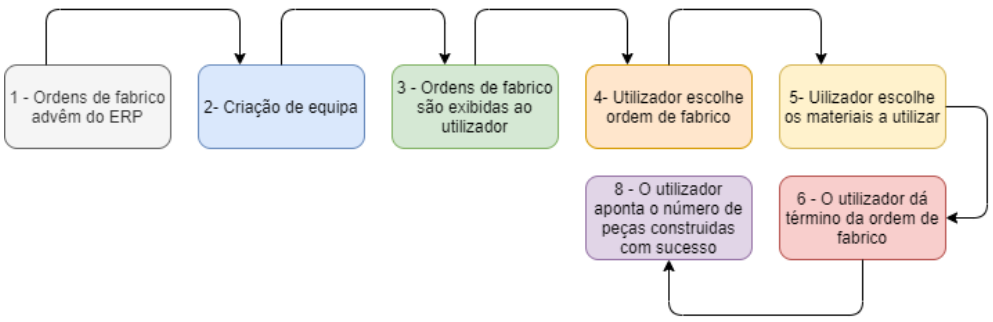

Figura 20 - Processo de funcionamento da execução de ordens de fabrico

Na Figura 20, pode-se observar todas as etapas que foram definidas no levantamento de requisitos para a elaboração desta aplicação. Na primeira etapa, referenciado na figura como "1 – Ordens de fabrico advêm do ERP" refere-se a um pedido realizado ao ERP para obter as ordens de fabrico mais recentes, partindo do pressuposto de que a fábrica possui um nível de digitalização suficiente para operar com um ERP.

A segunda etapa, referida como "2 – Criação de equipa", implica que, para executar uma ordem de fabrico, um ou mais colaboradores estão associados a essa tarefa. Sendo que a identificação das pessoas afetas a essa ordem de fabrico está associada a essa fase.

Na terceira fase "3 – Ordens de fabrico são exibidas ao utilizador", uma lista de todas as ordens de fabrico por executar é apresentada ao utilizador.

Na fase 4 – "4 – Utilizador escolhe ordem de fabrico", é escolhida a ordem de fabrico a ser executada pela equipa formada no passo 2.

Na fase 5 – "Utilizador escolhe os materiais a utilizar", é apresentada uma lista ao utilizador com todos os materiais afetos à ordem de trabalho selecionada anteriormente, e este tem de escolher a quantidade de cada material que necessita para a execução da ordem de fabrico. Após a fase 5 deste processo, a ordem de fabrico é dada como iniciada.

Na fase 6, identificada na figura como "6 – O utilizador dá término da ordem de fabrico", o utilizador aciona a paragem da máquina, sendo que o tempo produção/assemblagem é apontado automaticamente.

Na fase 7, identificada na figura como "8 – O utilizador aponta o número de peças construídas com sucesso", refere-se à verificação de que as peças estão ou não conformes, com a devida anotação das quantidades realizadas com sucesso ou não e finda-se o processo de execução de uma ordem de fabrico.

Também se ponderou sobre a natureza da aplicação a desenvolver, sendo que a *inCentea* já possuía uma aplicação semelhante em produção que era do tipo *desktop standalone*. Discutiu-se sobre a reformulação desta mesma aplicação, de forma a poder ser utilizada em dispositivos móveis e concluiu-se que a escolha mais óbvia seria não optar pelo *refactoring* (i.e., modificação da aplicação existente de forma a que esta consiga concretizar os requisitos funcionais e não funcionais), mas sim pela reformulação total desta aplicação num contexto mais móvel.

Optou-se por uma implementação de uma *web app* de forma a garantir assim o seu suporte à manutenção corretiva e evolutiva pela equipa de desenvolvimento *web* da *inCentea*.

Com uma aplicação *web*, este trabalho de manutenção é simplificado, exigindo também a quem o faça que domine menos linguagens de programação e não tenha uma curva de aprendizagem tão acentuada para se iniciar no projeto.

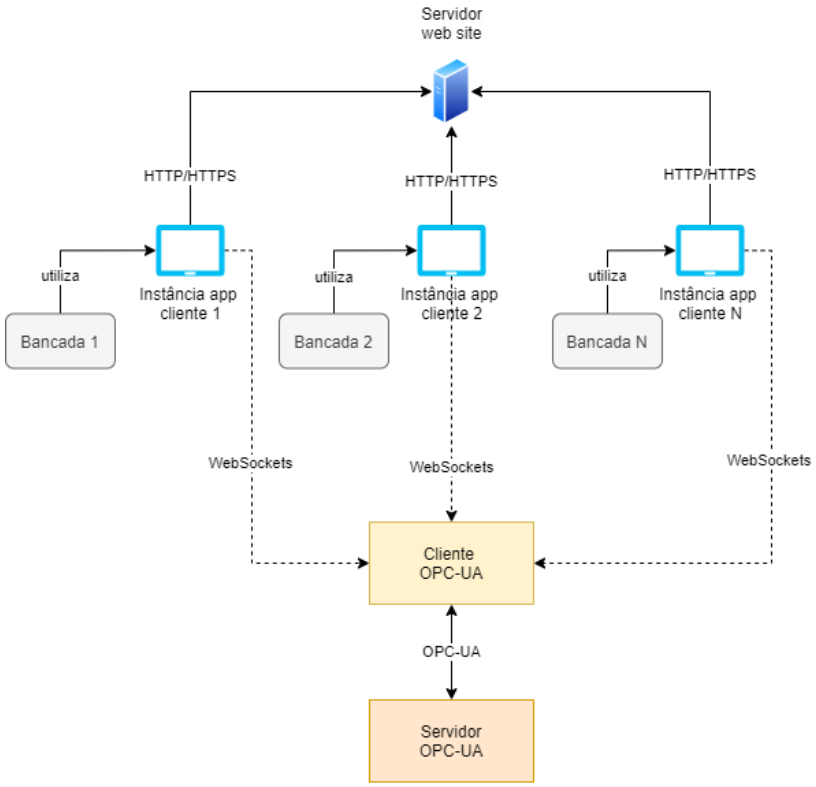

Figura 21 - Principais tecnologias utilizadas na integração entre OPC-UA e o serviço *Intelligent Molds*

A Figura 21 ilustra a arquitetura física idealizada para esta solução, exemplificando um cenário físico para a situação descrita na Figura 19. Sendo este cenário uma arquitetura clássica cliente-servidor. A aplicação *web* pode funcionar como uma solução *on-premise* (com o servidor dentro das instalações da fábrica, num cenário de *intranet*), ou em *cloud* – com o servidor alojado nas infraestruturas de um *cloud provider*, o que tem um custo acrescido associado à configuração da *firewall* da fábrica para suportar comunicação com a *Internet* e para a utilização de recursos do *cloud provider*, sendo que esta opção operacional será realizada pelo cliente e não implicando configurações adicionais na aplicação de forma a assegurar que o cenário aplicacional esteja funcional.

Numa fábrica poderão existir várias bancadas com vários funcionários a consumir a aplicação, sendo que a cada um estará afeto um dispositivo (e.g. *smartphone*, *workstation* ou *tablet*) e em cada um desses dispositivos existirá uma instância da *app* a correr.

A comunicação entre a *app* e o cliente OPC-UA está ilustrada na Figura 21 com setas em tracejado. Optou-se pela utilização da tecnologia *WebSockets* [44] nesta comunicação devido a este protocolo permitir a existência de canais de comunicação bidirecionais, o que é uma necessidade neste caso. Se se tivesse optado pela arquitetura cliente-servidor "pura", o servidor poderia servir pedidos "bloqueantes" porque este serve como uma *Facade* para o ERP, e no caso de a conetividade estar comprometida, a *app* também se tornará irresponsiva. Utilizando a tecnologia *WebSockets*, é criada uma sessão entre os dois pontos de comunicação, onde existe uma transferência de informação assíncrona bidirecional (i.e., pedido da parte da *app* cliente pelas ordens de fabrico, submissão da ordem de fabrico concluída), que permite, por instância, a notificação da parte do cliente OPC-UA à *app* quando receber as ordens de fabrico do ERP. O motivo pelo qual se utilizou a tecnologia *WebSockets* para esta comunicação por detrimento de OPC-UA foi devido à inexistência de bibliotecas que implementassem um cliente OPC-UA em *Angular* (*framework* utilizada para o desenvolvimento da *app*).

A *stack* tecnológica (conjunto de tecnologias a utilizar neste projeto), foi acordada com a *inCentea* de forma a tentar também integrar o projeto o máximo possível com o conjunto de competências da empresa. Decidiu-se, assim, que a arquitetura genérica desta aplicação uma *Single Page Architecture* (SPA), que permite oferecer ao utilizador uma experiência similar à que o utilizador teria se utilizasse uma aplicação *desktop* nativa. Esta opção de implementação foi tida em conta devido aos colaboradores que utilizam a aplicação obsoleta (*desktop* nativa), terem de passar por uma curva de aprendizagem para conseguirem operar esta *app*. Optou-se por uma SPA para tentar suprimir ao máximo essa resistência à mudança da parte do utilizador da *app*, resistência que poderia suceder-se no caso da aplicação não ter um desempenho ao nível de uma aplicação nativa.

Tendo em conta os desafios que uma recolha inteligente de dados pode apresentar, pensou-se como solução numa aplicação informática multiplataforma, para que pudesse ser utilizada pelos operadores através de um computador *desktop* fixo ou mesmo de um *tablet*.

De forma a tentar limitar o tempo de desenvolvimento de uma solução multiplataforma, optou-se pelo desenvolvimento de uma aplicação *web* (ao invés de, por exemplo, uma aplicação móvel nativa para dispositivos móveis), pois o desempenho *real-time* não era um requisito não-funcional de risco para esta solução. Tendo a *inCentea* uma solução similar em produção, optou-se por manter a base em termos de funcionalidades e fazer uma reformulação da *User Interface* (UI), sendo que desta forma seria possível ter um intervalo de confiança elevado de que a aplicação teria uma maior aceitação por parte dos operários fabris de chão de fábrica.

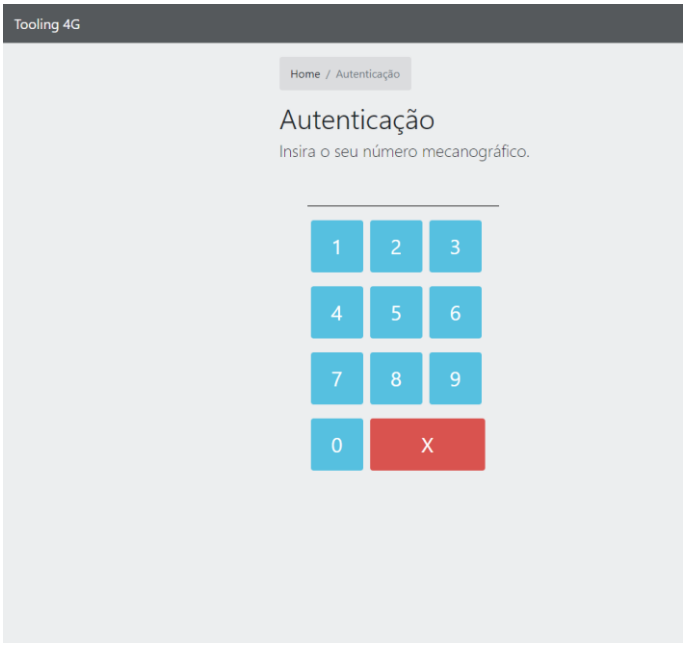

Figura 22 - Aplicação cliente de ordens de fabrico - ecrã de autenticação

Ao abrir a aplicação, o primeiro ecrã possibilita ao utilizador a inserção do seu número mecanográfico para efeitos de autorização no sistema (Figura 22). O tamanho dos botões foi propositadamente aumentado e espaçado, por forma a permitir o mínimo de cliques possíveis e consequentemente, a diminuir a taxa de erro na utilização da aplicação. Após o operário que estiver a introduzir o PIN colocar o seu número mecanográfico, a aplicação irá automaticamente, carregar a próxima página sem necessitar de um botão "Seguinte" para o efeito, de forma a diminuir o número de cliques necessários para a execução de uma tarefa.

No caso em que os números mecanográficos estão num cartão, associados a um código de barras ou código *QR*, é também possível, utilizando um leitor de códigos de barras/*QR Code*, associado ao dispositivo em utilização (*i.e.*, *tablet* ou *desktop*), a inserção do número neste mesmo ecrã via *scanner* de código de barras, sem ter de o introduzir manualmente.

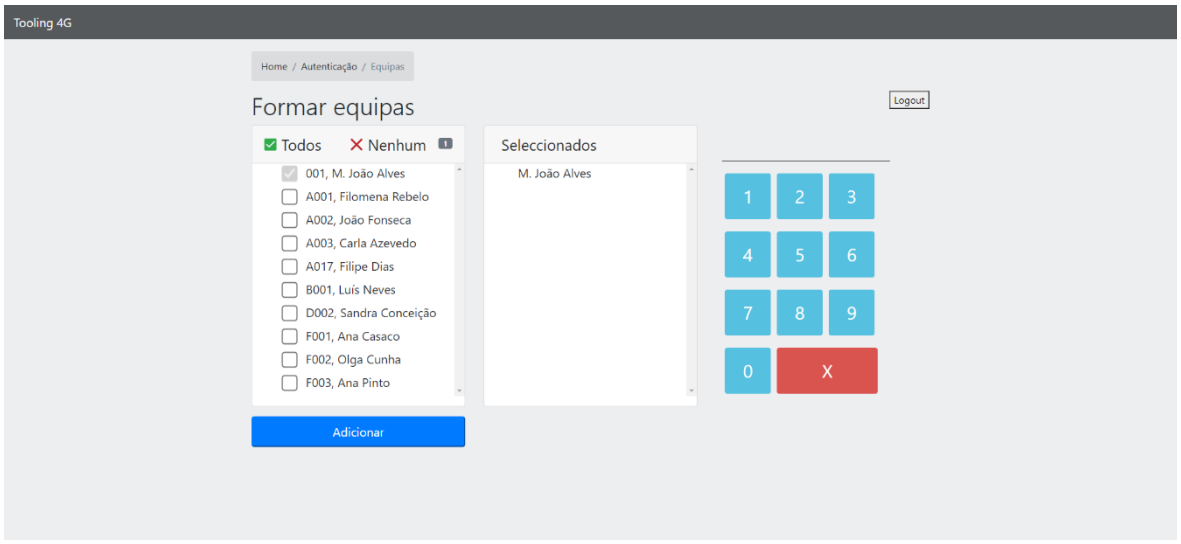

Figura 23 - Aplicação cliente de ordens de fabrico - ecrã de formação de equipas

No segundo passo do processo de seleção de uma ordem de trabalho, após o processo de autorização na página anterior, o programa necessita de conhecer quais os colaboradores que irão participar na "equipa" para a execução da ordem de fabrico (Figura 23). Para tal, existem vários métodos de inserção de dados. Sendo que o primeiro e mais óbvio, é a seleção do nome do colaborador na lista à esquerda. Também é possível adicionar um elemento à equipa utilizando o teclado virtual do lado direito para introduzir o número mecanográfico do mesmo. Da mesma forma, também se pode ler o código de barras/*QR Code* do funcionário com um leitor, à semelhança como acontece na página anterior, de forma a associar esse mesmo funcionário à equipa, para a execução da ordem de trabalho.

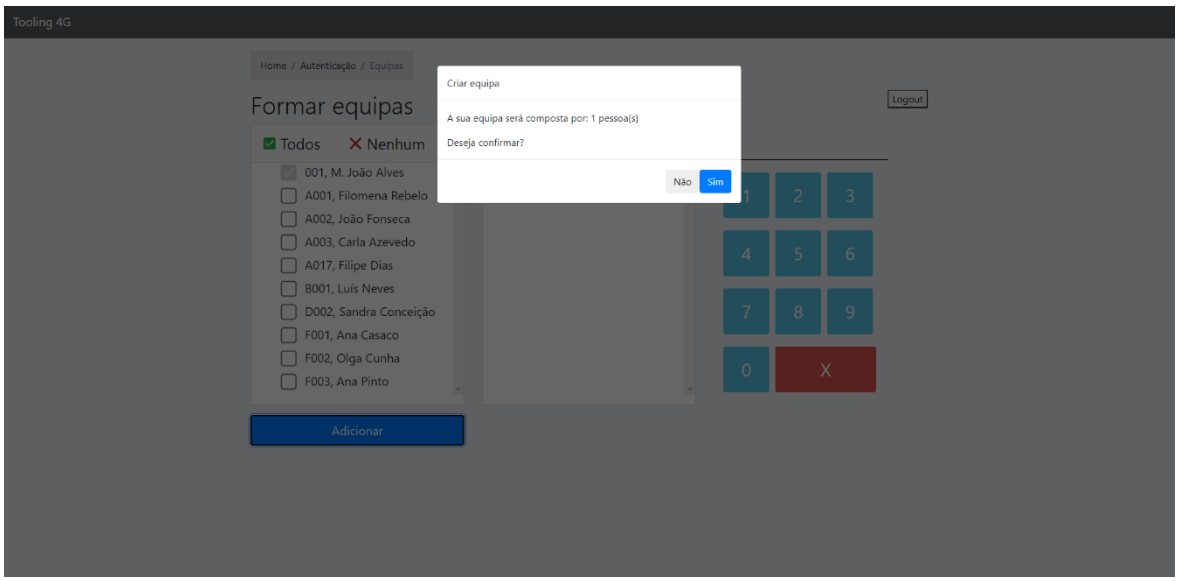

Figura 24 - Aplicação cliente de ordens de fabrico - ecrã de confirmação da formação de equipas

Após a seleção da equipa, a aplicação irá pedir a confirmação – Figura 24 – de forma a validar se a ação tem algum erro de *input* antes de se proceder ao passo seguinte no processo de seleção da ordem de trabalho.

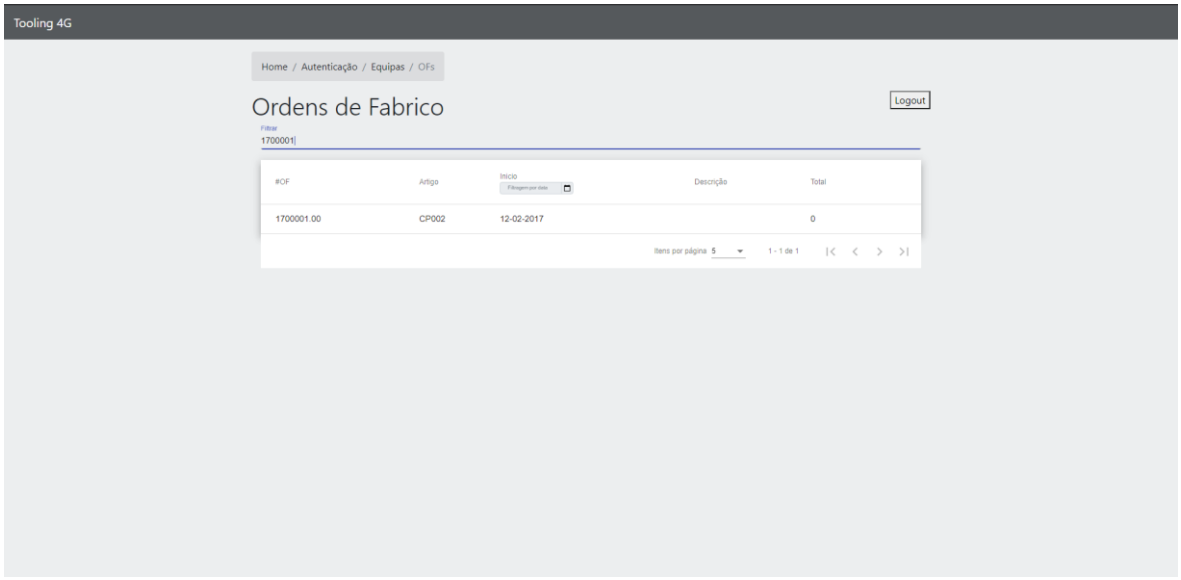

Figura 25 - Aplicação cliente de ordens de fabrico - ecrã de listagem de ordens de fabrico

Tendo a equipa escolhida, o próximo passo é a seleção da ordem de trabalho Figura 25. Para tal, a aplicação irá mostrar a lista de todas as ordens de trabalho a realizar em forma tabular. Na forma de visualização tabular torna-se mais simples de mostrar um grande volume de informação relacionada, além de que permite ordenações e filtragens com poucos cliques, sendo este o motivo de se ter optado por uma tabela nesta página e em páginas similares.

Nesta página, é possível ordenar as ordens de trabalho por ordem alfabética ou alfanumérica, ascendente ou descendente, clicando na coluna em que se pretende ordenar – No caso de existirem muitas ordens de trabalho, para que o operador não perca muito tempo à procura da que pretende executar. Também é possível filtrar por data de início, clicando no calendário e escolhendo se deseja ver as ordens de trabalho do dia atual ou do dia anterior. Não se colocou um *date picker* porque tal iria implicar mais cliques e iria acrescentar complexidade ao processo, sendo que após reunião com o gestor de equipa da inCentea, se decidiu que em 90% dos casos se utilizam estes dois filtros em aplicações que já estão a ser utilizados por clientes, em ambiente de produção.

À semelhança dos ecrãs anteriores, se existir um leitor de códigos de barras ou *QR Code*, é possível utilizá-lo para ler o código associado à ordem de trabalho no campo “Filtrar” do referido ecrã.

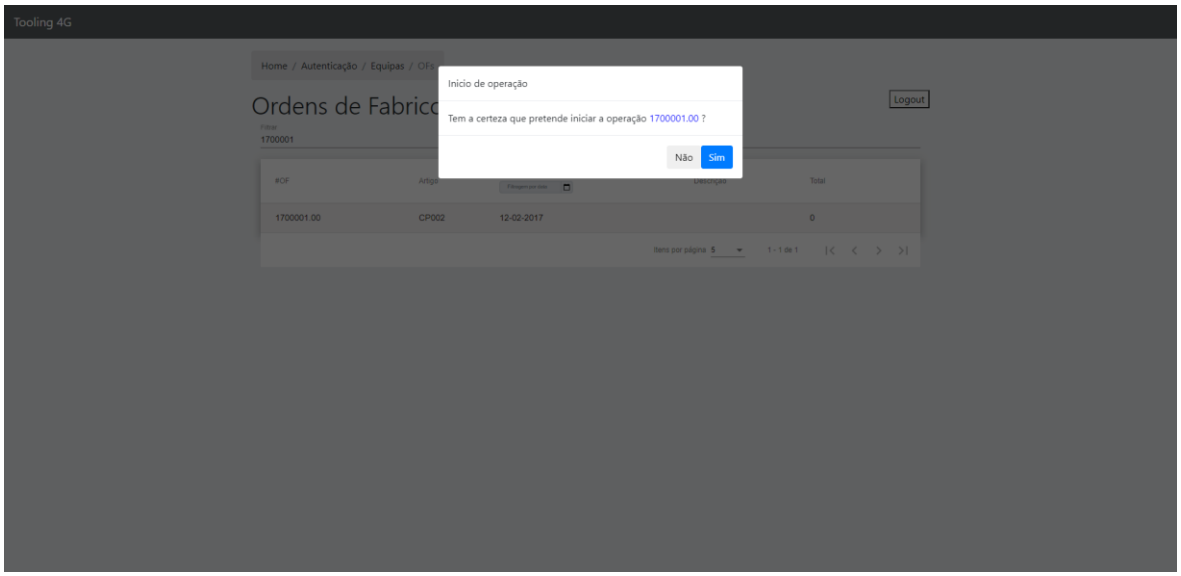

Figura 26 - Aplicação cliente de ordens de fabrico - ecrã de confirmação de escolha da ordem de fabrico

Após clicar na ordem de trabalho pretendida, como se pode observar na Figura 26, o sistema irá validar com o utilizador se pretende iniciar a ordem de trabalho clicada de forma a prevenir possíveis erros.

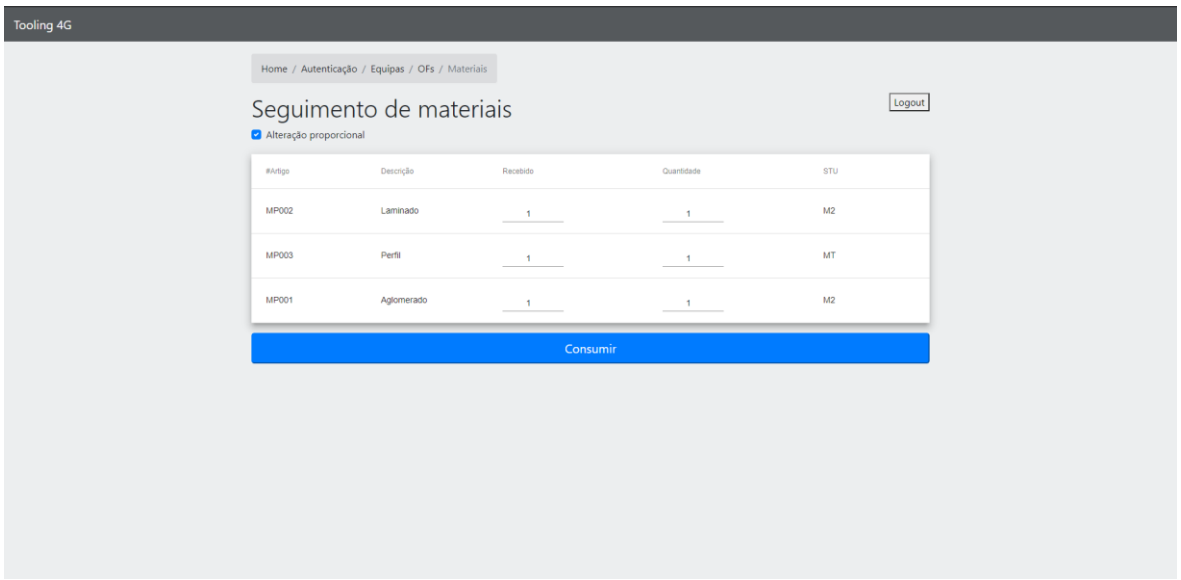

Figura 27 - Aplicação cliente de ordens de fabrico - ecrã de listagem de materiais e quantidades associadas à OF

O passo seguinte é o de seleção do número de materiais recebidos e a quantidade a ser utilizada na execução da ordem de trabalho (Figura 27). Após o utilizador clicar no botão "Consumir", irá aparecer novamente uma página de confirmação da ação de forma a alertar o utilizador para a inserção de possíveis erros de *input*.

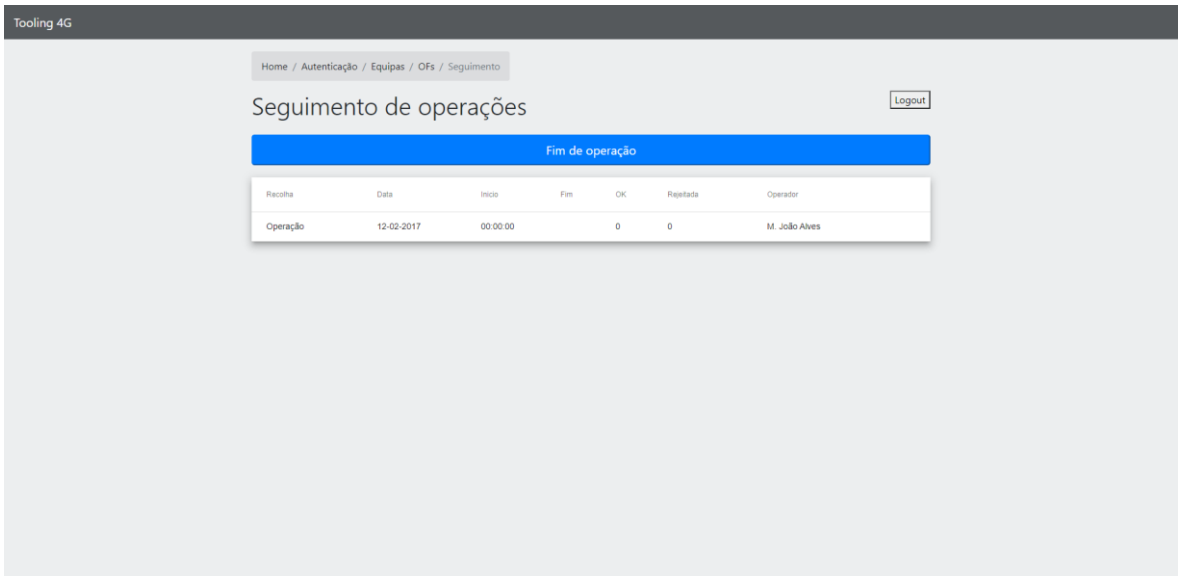

Figura 28 - Aplicação cliente de ordens de fabrico - ecrã de seguimento de operações

No ecrã de seguimento de operações (Figura 28), os utilizadores têm acesso à ordem de trabalho que se encontra ativa, sendo que a cada linha nesta tabela corresponde à tarefa que cada funcionário está a executar associada à ordem de trabalho em questão. Para dar o término de todas as tarefas, basta clicar no botão "Fim de operação".

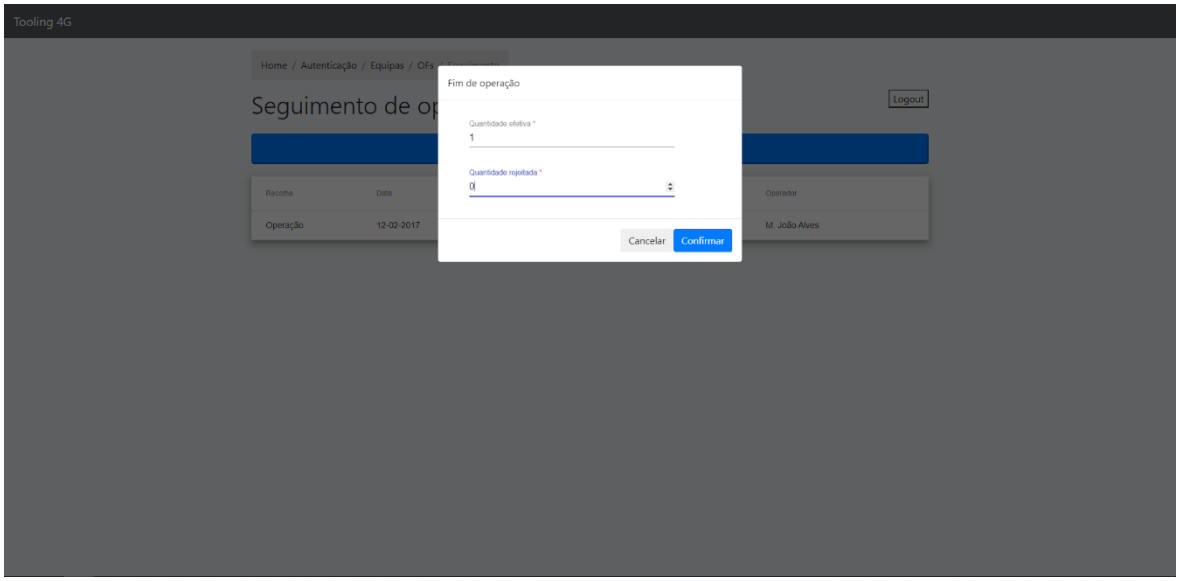

Figura 29 - Aplicação cliente de ordens de fabrico - ecrã de confirmação de quantidades com defeito ou bem produzidas

Ao clicar no botão "Fim de operação", inicia-se o processo de validação em que o utilizador colocará o número de peças produzidas com sucesso e o número de peças que foram rejeitadas (Figura 29). O sistema irá calcular o tempo despendido na ordem de trabalho e irá enviar para o sistema de informação de gestão da empresa (i.e. ERP).

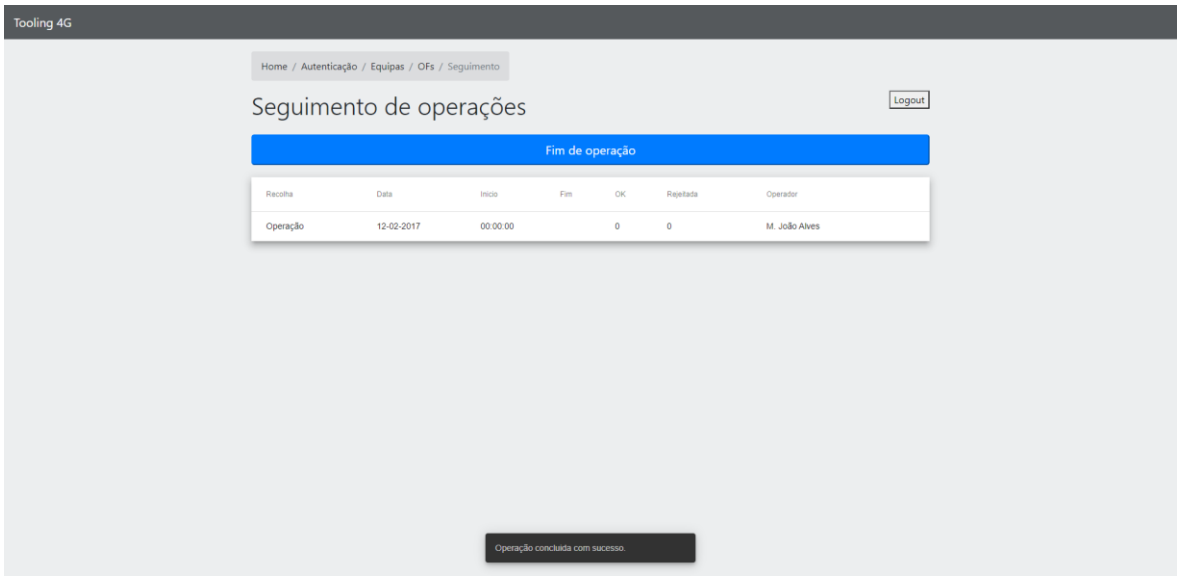

Figura 30 - Aplicação cliente de ordens de fabrico - ecrã de alerta de ordem de fabrico realizada com sucesso

Por fim, a aplicação tentará comunicar com este sistema utilizando a rede OPC-UA para esse efeito. Se existir algum problema de conetividade, o utilizador será alertado desse evento e poderá mais uma vez tentar findar o processo. Se não existirem problemas técnicos ao comunicar com o ERP, o utilizador irá visualizar a mensagem exibida na Figura 30.

A recolha de *inputs* num chão de fábrica, de forma a não prejudicar o desempenho do processo de produção, é um assunto complexo e a proposta de solução apresentada revela-se como um sistema "inteligente" tanto quanto a compatibilidade de periféricos de entrada, como no que respeita ao funcionamento em diferentes dispositivos. Sendo uma aplicação *web* responsiva, a mesma pode ser utilizada em diferentes resoluções de ecrã – seja telemóveis, *tablets*, *iPads* e *desktops.* Esta pode ser operada utilizando um dispositivo com ecrã *touchscreen*, teclados virtuais, teclados físicos ou mesmo *scanners* de códigos de barra e/ou *QR Codes* ligados ao dispositivo e sem precisar de nenhuma configuração para o efeito. Um cuidado acrescido foi tido no desenho da *User Experience* (UX), de forma a minimizar o número de cliques por ação pretendida e o número de erros por ação a realizar, entre outras métricas. O facto de esta solução ser um redesenho de *interface* gráfica de uma aplicação já existente e testada em ambientes de produção industriais traduz-se em confiança acrescida na usabilidade da mesma.

5. Testes

A utilização de uma pilha de testes de *software*, podendo estes testes serem automáticos ou manuais, permite reforçar a confiança por parte de um *stakeholder* de que um projeto terá menos erros de programação (i.e., *bugs*), bem como possibilita uma maior velocidade na sua respetiva correção evolutiva. A utilização de uma fase de testes numa metodologia de desenvolvimento de software (seja esta ágil, ou de modelo em cascata), permite a criação de um produto mais confiável, bem como a deteção e correção de erros numa fase precoce (se a fase de testes for corretamente implementada na respetiva metodologia de desenvolvimento).

Algumas *frameworks* de testes de aceitação executam o código da aplicação a testar, sendo ignorantes quanto à forma como este está escrito, tratando-o como uma caixa preta. A execução de testes também é conhecida como análise dinâmica de código, exatamente pelo motivo de o código ter de ser executado para se conseguir testá-lo.

A exceção da análise dinâmica de código, é a análise estática, em que não é necessário executar o código para afirmar o seu grau de assertividade. A análise estática de código é um método que permite afirmar com um certo grau de confiança, se um repositório de código é assertivo, tendo em conta vários fatores – *bugs* ou pedaços de código com comportamentos não desejados propositadamente pelo programador; *code smells* que consistem em más práticas de programação que inviabilizam uma correção evolutiva do repositório de código e propiciam a introdução acidental de *bugs*; erros de desempenho que podem tornar a execução do código mais lenta; erros de segurança que podem possibilitar roubo de informação, problemas de integridade dos dados ou de confidencialidade, entre outros.

5.1. Análise estática de código

O processo de análise estática de código pode ser inserido em qualquer metodologia de desenvolvimento de *software*, sendo que é mais comumente encontrada em metodologias de desenvolvimento ágil devido à fácil integração destas ferramentas automatizadas numa *pipeline Continuous Integration*/*Continuous Delivery* (CI/CD), sendo normalmente realizado antes da execução dos analisadores dinâmicos de código (testes automatizados).

Com base nas premissas referidas da análise estática de código, decidiu-se optar pela utilização de uma destas ferramentas no desenvolvimento do projeto, sendo que a ferramenta escolhida, por uma questão de já existir experiência de utilização anterior, foi o *SonarQube* [45].

SonarQube é uma ferramenta de análise estática de código aberto, desenvolvida pela empresa *SonarSource*. Sendo um software com compatibilidade no sistema operativo *Windows* e no *Ubuntu* e variantes Linux.

Sendo que na sua maioria, o projeto *Tooling 4G* foi escrito na linguagem C#, utilizando o *Integrated Development Environment* (IDE) *Visual Studio*, num sistema operativo Windows, optou-se pela utilização de um *container Docker* para separar este processo, também de forma a torná-lo mais portátil entre diferentes sistemas.

O *SonarQube* foi integrado no ambiente de desenvolvimento utilizando o *Docker*, criando um *container Linux* com o servidor *SonarQube* instalado e configurado. A vantagem da utilização de um *Docker container* para este propósito prende-se com o princípio da responsabilidade única (*single responsability principle*) e portabilidade. O princípio da responsabilidade única refere que uma entidade deve ter um só objetivo. Sendo este um princípio afeto principalmente ao paradigma de programação orientado a objetos (POO), também se pode extrapolar para outos assuntos como neste caso, em que ao se restringir as ferramentas de desenvolvimento em *containers*, se evita a incompatibilidade de versões de dependências, bem como se cria um ponto singular de alteração no caso de mudanças no ambiente de desenvolvimento ou nos requisitos do projeto. A utilização de *Docker containers* neste âmbito também possibilita uma maior inter-compatibilidade entre ambientes de desenvolvimento e sistemas operativos, devido ao *Docker engine* (i.e., motor que transforma os pedidos dos *Docker container* em *system calls* para a *kernel* do sistema operativo anfitrião) possuir compatibilidade com os principais sistemas operativos do mercado – *Windows*, *Linux*, *BSD* e *macOS X*.

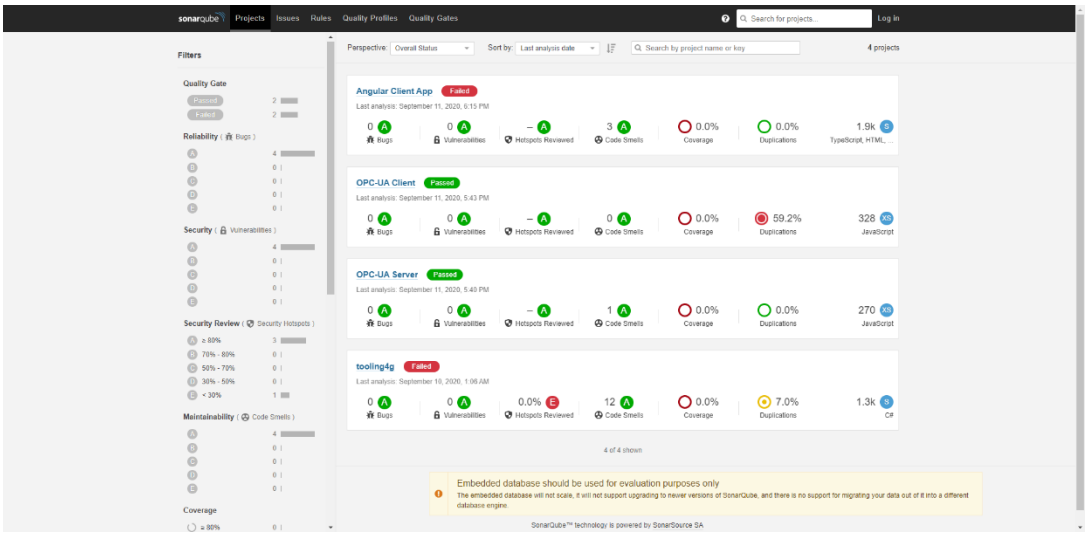

Figura 31 - Listagem de todos os projetos e estatisticas do *SonarQube*

Na Figura 31 pode-se observar o painel administrativo do *SonarQube* com estatísticas para todos os projetos associados à solução *Tooling 4G*. Nesta página pode-se observar métricas – como o número de *bugs*, vulnerabilidades de segurança, os já referidos *code smells*, código duplicado e métricas referentes a testes unitários – nomeadamente a *code coverage* que consiste na quantidade de código testado, que os testes unitários conseguem cobrir.

Como ilustrado no exemplo da Figura 31, e dando o breve exemplo da aplicação *OPC-UA Server*, só existe um *code smell*, e 270 linhas de código na linguagem *JavaScript*. No caso de um dos projetos não reunir os critérios de qualidade definidos no *SonarQube*, então irá aparecer um ícone a vermelho (ao lado do nome do projeto) "*Failed*" além de aparecer um ícone a vermelho num dos parâmetros analisados (e.g. *bugs*, vulnerabilidades, entre outros) a indicar qual ou quais dos parâmetros fizeram o projeto "chumbar".

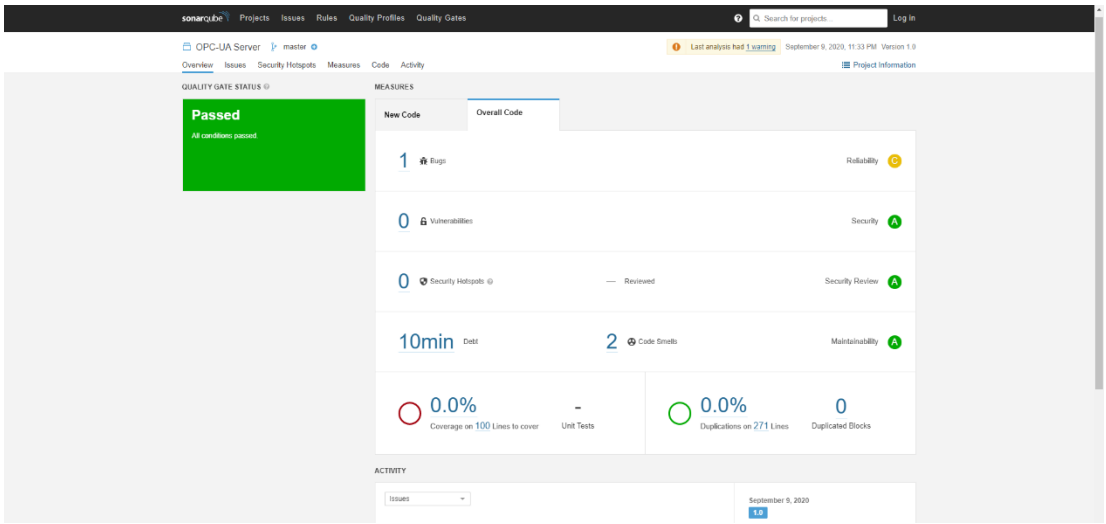

Figura 32 - Estatísticas relevantes ao projeto *OPC-UA Server* no *SonarQube*

Na Figura 32 pode-se observar o ecrã que se obtém ao clicar num dos projetos exibidos na imagem anterior. Neste ecrã, é possível obter detalhes mais avançados relativamente a um determinado projeto bem como o débito técnico – quantidade de esforço técnico na correção de um problema, medido em unidades de tempo – necessário para a solução dos problemas de qualidade.

5.2. Análise dinâmica de código

Foram também desenvolvidos diferentes tipos de testes automáticos de forma a garantir a confiança na entrega do produto aos *stakeholders*. A abordagem na metodologia de testes foi puramente holística, sendo que existiu uma dispersão no tipo de testes tentando englobar os pontos que se determinaram mais críticos no que toca aos requisitos da aplicação. Nenhum dos tipos de testes foi realizado de forma exaustiva, daí a não utilização de métricas quantitativas, como por exemplo o *branch coverage* ou o *code coverage*, existindo um maior foco em métricas qualitativas referentes a "*test questions*" que foram propostas pelo autor:

1. O código possui o mínimo suficiente de qualidade para ser considerado um produto e consequentemente facilitar-se uma manutenção evolutiva?
2. A *user interface* permite a execução dos requisitos funcionais?
3. Sendo o *Tooling 4G* uma solução com uma arquitetura orientada ao serviço, a comunicação inter-serviço funciona?
4. Existindo um número *x*, que corresponde ao maior número possível de utilizadores que podem utilizar a *app* em simultâneo, a *app* conseguirá prestar os serviços mínimos que assegurem o cumprimento dos requisitos funcionais da mesma?
    a. Ainda referente à pergunta anterior – terá um Sistema de Gestão de Base de Dados (SGBD) relacional, um desempenho superior comparativamente a um SGBG não-relacional?

Ao observar as "*test questions*" acima propostas, pode-se identificar rapidamente os tipos de testes a serem desenvolvidos para as responder – análise estática de código, testes de aceitação da *interface* de utilizador, testes de integração e testes de volume e *stress* (desempenho).

1 – O código possui o mínimo suficiente de qualidade para ser considerado um produto e consequentemente facilitar-se uma manutenção evolutiva?

A qualidade do código, no que toca a *bugs*, duplicação de código, problemas relativamente a segurança e *code smells* foi assegurada com a utilização de analisadores estáticos – como foi referido no subcapítulo anterior – onde para cada *build* da solução, se executava o *software SonarQube* e se verificava se o código se encontrava em conformidade para seguir para a fase de pré-produção.

2 – A *user interface* permite a execução dos requisitos funcionais?

A *user interface*, sendo o ponto de entrada para toda a interação com o utilizador, torna-se um objeto de teste interessante, principalmente devido a ser esta que permite a execução dos requisitos funcionais pensados para a solução. Posto isto, e tendo a lista de requisitos funcionais sido apresentada no capítulo 3.1.1, foram desenvolvidos vários testes de aceitação de *user interface* com base nestes mesmos requisitos.

3 – Sendo o *Tooling 4G* uma solução com uma arquitetura orientada ao serviço, a comunicação inter-serviço funciona?

Numa *Service Oriented Architecture* (SOA), como é o caso no *Tooling 4G*, cada serviço funciona de uma forma semi-independente, tendo o seu próprio processo designado pelo sistema operativo e possuindo o seu respetivo *address space.* Sendo que os serviços dependem indiretamente uns dos outros no que toca a dados, a sua comunicação é um dos pontos fulcrais que permitem o asseguramento dos requisitos funcionais aos utilizadores e posto isto, decidiu-se implementar testes de integração que verifiquem a conetividade entre serviços desta mesma solução.

4 – Existindo um número *x,* que corresponde ao maior número possível de utilizadores que podem utilizar a *app* em simultâneo, a *app* conseguirá prestar os serviços mínimos que assegurem o cumprimento dos requisitos funcionais da mesma?

Sendo a natureza do projeto *Tooling 4G* uma aplicação distribuída, é expetável que existam vários colaboradores que utilizem a aplicação em simultâneo. Até dado o facto de que esta aplicação irá incidir sobre o chão de fábrica de uma fábrica de moldes. Posto isto, é necessário entender se a aplicação possui o desempenho necessária para gerir a carga e o *stress* imposto pela utilização de forma a assegurar a disponibilidade.

Para entender os limites de carga e stress a aplicação consegue gerir, foram realizados testes de *stress* e desempenho, tendo em conta um número ótimo de utilizadores com base no número de funcionários de um chão de fábrica de uma fábrica de moldes.

4. a) - Terá um Sistema de Gestão de Base de Dados (SGBD) relacional, um desempenho superior comparativamente a um SGBG não relacional?

Todos os serviços podem ter um nível ótimo de desempenho, mas sendo que todos estes serviços têm uma dependência com o repositório onde a informação está armazenada – a base de dados – esta pode criar um ponto singular para problemas de desempenho. De forma a suprimir este risco, foi desenvolvido um estudo que visa realizar um *benchmarking* entre os dois tipos de Sistemas de Gestão de Bases de Dados (SGBD) mais utilizados do mercado – SQL e NoSQL – e a respetiva compilação desses mesmos dados.

Testes de integração

Os testes de integração, que certificam que parte da comunicação entre serviços tem o *output* esperado dado um certo *input*, foi conseguido utilizando para efeito a *framework NUnit* [46].

Todos os *endpoints* relativos à API que comunicam com a interface gráfica foram testados de forma a verificar se estes devolvem o resultado expetável seja na *happy path* – inserindo *inputs* corretos – ou dando *inputs* errados.

Testes de aceitação da *user interface*

À semelhança do que aconteceu com os testes de integração, utilizou-se instrumentação semelhante para realizar os testes de aceitação da *user interface* – a *framework* de testes *NUnit*, em conjunto com a ferramenta *Selenium* que permite recriar as principais ações realizadas nos *browsers* mais utilizados do mercado – *Internet Explorer*, *Edge*, *Chrome*, *Firefox* – e permite a criação de testes robustos de aceitação utilizando a interface gráfica da aplicação para esse efeito. Todos os passos a executar nos testes de aceitação foram auxiliados pela ferramenta *Katalon Recorder* que consiste numa extensão para os *browsers* modernos (utilizou-se o *Google Chrome* para este efeito) e grava todas as ações que são efetuadas no mesmo, em formato *Selenium*, permitindo também a exportação em código fonte nas várias linguagens de programação e *frameworks* de testes mais utilizadas. Neste caso a exportação foi realizada na linguagem C# e para a *framework NUnit*.

Os objetos de teste neste âmbito são as duas aplicações com *interface gráfica* – o *dashboard* dos OEE e a aplicação de gestão de ordens de fabrico para o chão de fábrica.

Testes de carga

De forma a aferir o bom comportamento da aplicação sobre stress, foram realizados testes de carga. Nestes testes que também foram contextualizados com a pergunta de teste acima referida – "Existindo um número *x*, que corresponde ao maior número possível de utilizadores que podem utilizar a app em simultâneo, a app conseguirá prestar os serviços mínimos que assegurem o cumprimento dos requisitos funcionais da mesma? – sendo *x* um valor desconhecido, associado ao número de acessos simultâneos em paralelo, optou-se por utilizar um número razoável de 500 acessos em simultâneo.

Tendo-se atribuído o valor de 500 à constante *x*, foi criado um ambiente local de testes com um computador portátil com o sistema operativo *Windows 10 Home Edition*, um processador Intel Core i7 10510U, 16 GiB de *Random Access Memory* (RAM), e um *Solid State Drive* (SSD) SAMSUNG MZVLB1T0HALR como dispositivo de persistência. O computador foi colocado no modo de desempenho no que toca à gestão de energia do sistema operativo e o *software Apache JMeter* foi utilizado para a criação dos testes de carga.

Apache JMeter [47] é um software para testes de carga desenvolvido e distribuído pela *Apache Software Foundation* com a licença *Apache 2.0*. Tem como dependência o Java na versão 8 – para a versão 5.3, utilizada neste ambiente de testes – este software foi utilizado para a criação dos testes de carga, empregando a *JMeter Graphic User Interface* (GUI) e para efeitos de *benchmarking*, foi utilizado o *JMeter Command-line* (CLI) *Mode* de forma a não criar mais *stress* desnecessário no ambiente de teste e poder de certa forma criar um viés negativo no resultados dos testes.

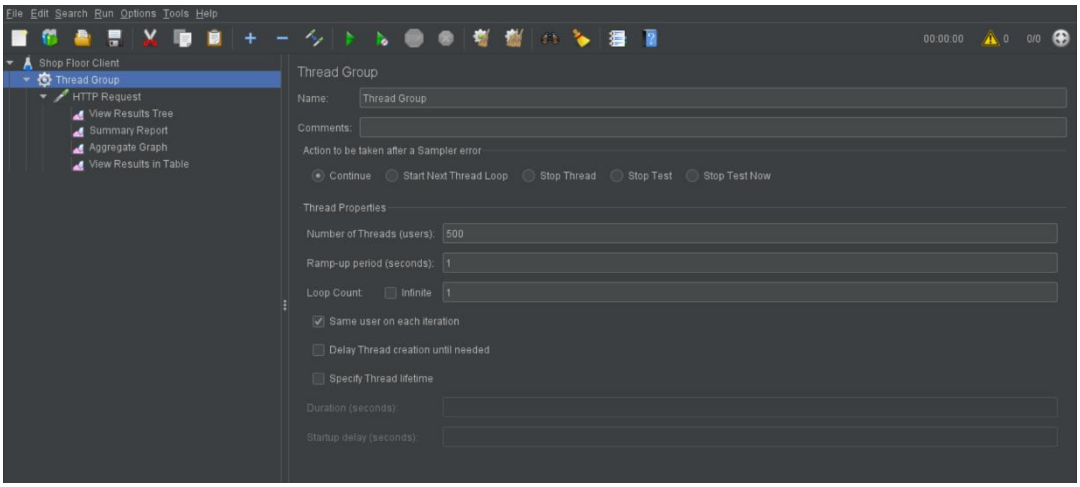

Figura 33 - Configurações dos testes de carga no *Apache JMeter*

Na Figura 33, pode se observar um *printscreen* do *Apache JMeter GUI*, nas definições da *Thread Group – JMeter* atribui uma *thread* para a simulação de um só utilizador – sendo que para este caso foram utilizadas 500 *threads*, com um período *ramp-up* de 1 segundo. O termo *ramp-up* consiste no tempo em segundos que o *JMeter* vai demorar para associar todos os utilizadores de teste (*threads*) à instância do teste.

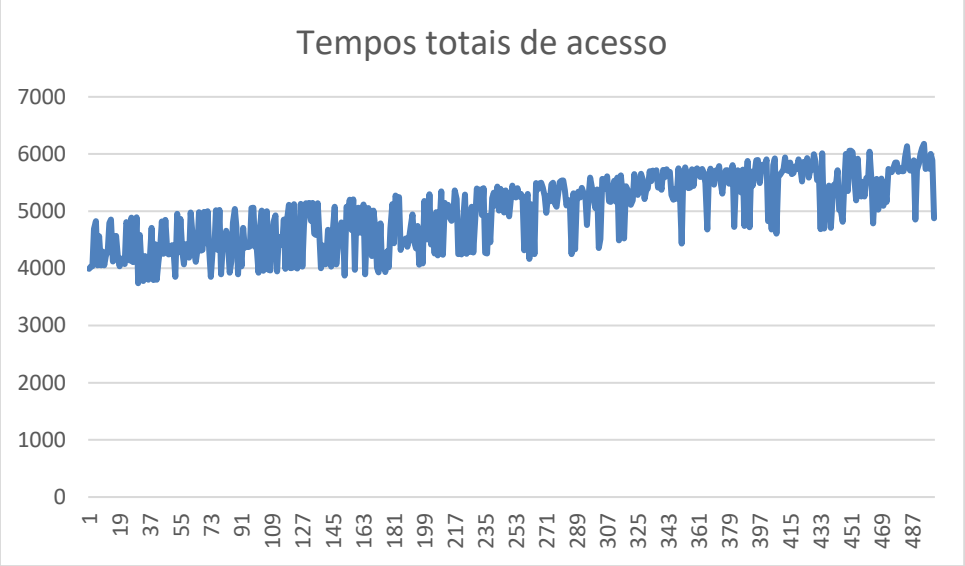

Figura 34 - Tempos totais de acesso simultâneo de 500 utilizadores na aplicação

Na Figura 34, pode-se observar um gráfico de linhas que representa os tempos totais de acesso (em milissegundos) no eixo das ordenadas de cada *thread* individual (eixo das abscissas), identificadas pelo seu respetivo número (e.g. *thread* 1, *thread* 2, etc.).

Pode-se observar que os tempos de acesso se fixaram entre os 3740 milissegundos (valor mínimo) e os 6178 (máximo), com um tempo médio de acessos de 5 segundos (5000 milissegundos).

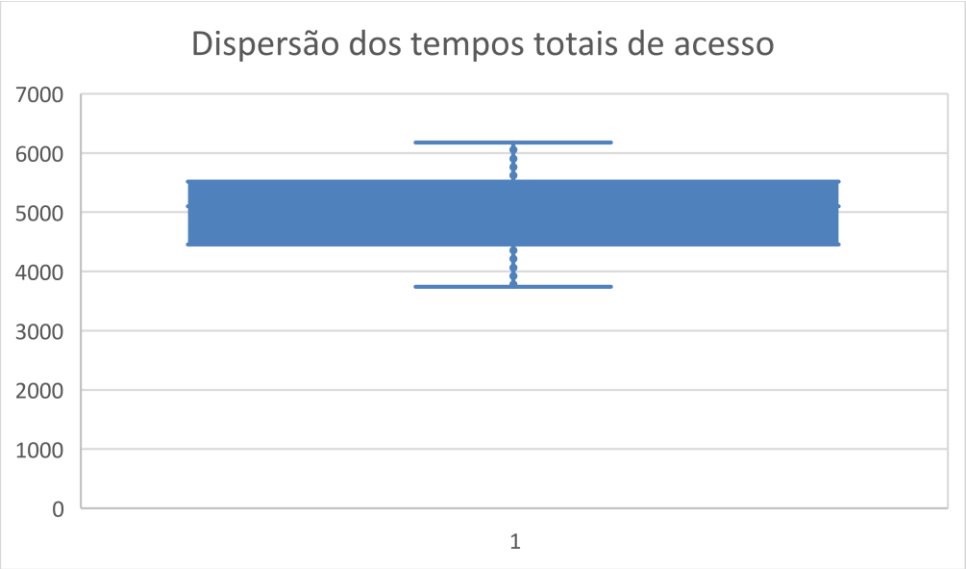

Figura 35 - Gráfico de dispersão dos tempos totais de acesso

A Figura 35 demonstra um gráfico *box plot* ou de dispersão onde se pode observar a disparidade dos dados, correspondendo estes aos tempos de acesso mais rápidos e mais lentos. Fixando-se a média nos 5000 milissegundos, o desvio padrão dos tempos de acessos foi de aproximadamente 606 milissegundos. No geral, pode-se concluir que não existiu uma dispersão significativa nos tempos de acesso, mesmo tendo em conta o ambiente de testes não ser o melhor.

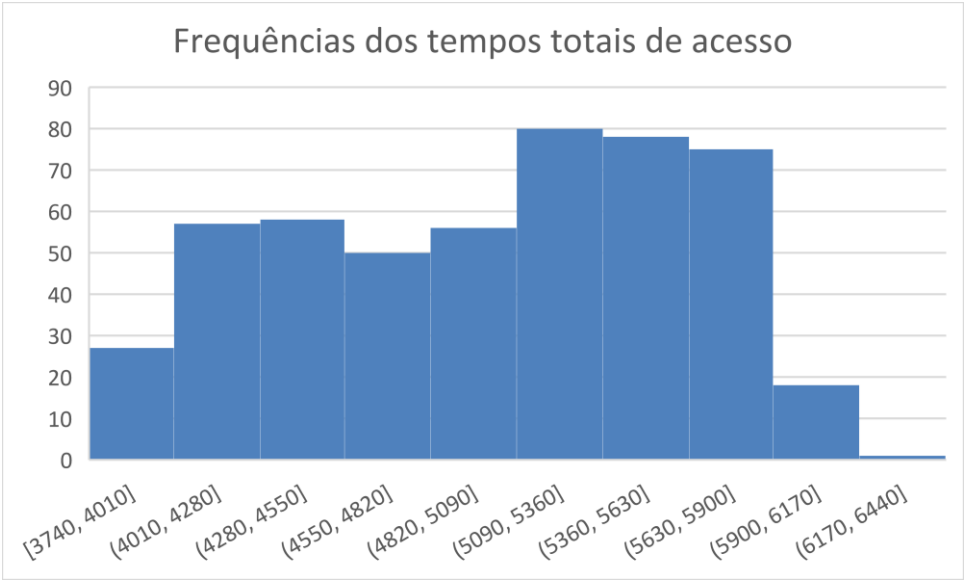

Figura 36 - Histograma com as frequências dos tempos totais de acesso

Na Figura 36, pode-se observar um histograma das frequências dos tempos totais de acesso. Sendo que no eixo das ordenadas observa-se a quantidade de *threads* e no eixo das abscissas os tempos de acesso, discretizados por classes.

Pode-se observar por este gráfico, que o valor mínimo (menos de 5 *threads*) está associado a tempos de acesso na ordem dos 6170 milissegundos até aos 6440 milissegundos, enquanto os que têm uma maior frequência estão associadas à classe que engloba tempos de acesso dos 5090 milissegundos aos 5360 milissegundos, com 80 *threads* entre as 500 utilizadas a atingirem estes tempos.

Tendo em conta que a frequência associada aos tempos de acesso mais morosos foi a que teve a frequência menor, pode-se concluir com este gráfico que o desempenho da *app* em termos de tempo de acesso se situa confortavelmente na ordem dos 5 segundos, no cenário associado ao pior caso (500 utilizadores acedendo simultaneamente).

Testes de desempenho SQL vs NoSQL

A necessidade do estudo comparativo de desempenho de um repositório relacional e de um repositório não-relacional deve-se devido ao grande volume de dados produzido num chão de fábrica devido ao diversificado número de sensores existentes no mercado, indicadores de desempenho, registos de atividades produzidas entre outros dados.

Tipicamente o modelo de dados de um chão de fábrica é altamente relacional, sendo que normalmente as entidades provenientes do modelo de dados têm relações muito fortes entre si, o que justifica a utilização de modelos relacionais por parte dos *Manufacturing Execution System* (MES) e também de *Enterprise Resource Planning* (ERP).

Em contrapartida, tem existido uma crescente adoção dos paradigmas *NoSQL*, orientados ao documento e não à relação, por existirem diversos estudos que apontam que este tipo de repositórios são justificáveis para cenários com um grande volume de dados (*Big Data*) e também para escritas mais rápidas [48].

De forma a realizar uma comparação entre estas duas diferentes tecnologias, foi realizado uma simulação do modelo de dados, existente na aplicação *OEE Web Server* (*dashboard* dos indicadores OEE), bem como realizado um *benchmark* (*stress/loading test*) para verificar as diferenças de desempenho de leitura e escrita relativamente a estes dois tipos de repositórios de dados.

Foram utilizadas as tabelas *Process* e *ProcessPiece*, sendo que a *ProcessPiece* tem 18 atributos e é a tabela onde são armazenadas todas as peças produzidas no âmbito de uma ordem de fabrico.

Sendo que estas tabelas já existiam numa base de dados *SQL Server* devido ao desenvolvimento do *OEE Dashboard*, optou-se então pela replicação destas numa base de dados *MongoDB* (*NoSQL*).

Relativamente à infraestrutura onde os testes foram corridos, estes foram executados num computador portátil com um processador *Intel Core i7 4700-HQ*, com 16 GiB de RAM e um disco rígido mecânico de 7200 rpm.

De forma a tentar manter uma *interface* que permitisse testar as duas bases de dados de igual forma, mesmo ambas terem formas distintas de funcionamento, optou-se pela criação de dois *web services* REST, que apenas disponibilizavam uma interface de leitura e de escrita para os repositórios e também um *Object Relational Mapping* (ORM), de forma a tornar o teste o mais imparcial possível. Como ferramenta de *load testing* foi utilizado o *Apache JMeter*.

O *web service* foi desenvolvido utilizando a *framework ASP .NET Core* na versão 2.1. e o ORM utilizado foi o *Entity Framework Core*.

Para determinar qual dos dois repositórios é o indicado para uma aplicação desta natureza, foram realizados dois testes onde foram registadas várias métricas.

Foram realizados dois testes a cada repositório, de leitura e escrita, respetivamente.

Para os testes de leitura, foram carregados 500 registos em cada uma das bases de dados, para os testes de escrita foram inseridos registos novos simulando um volume de escrita acima da média a que estes repositórios iriam provavelmente ser submissos.

As métricas utilizadas para medir o desempenho das duas bases de dados foram:

- Número de amostras (número de utilizadores, admitindo desde 50 utilizadores com 10 em simultâneo até 150 utilizadores com 30 em simultâneo);
- Desvio padrão do tempo de resposta dos pedidos em milissegundos;
- *Throughput* (número de pedidos por minuto);
- Média do tempo de resposta dos pedidos em milissegundos.

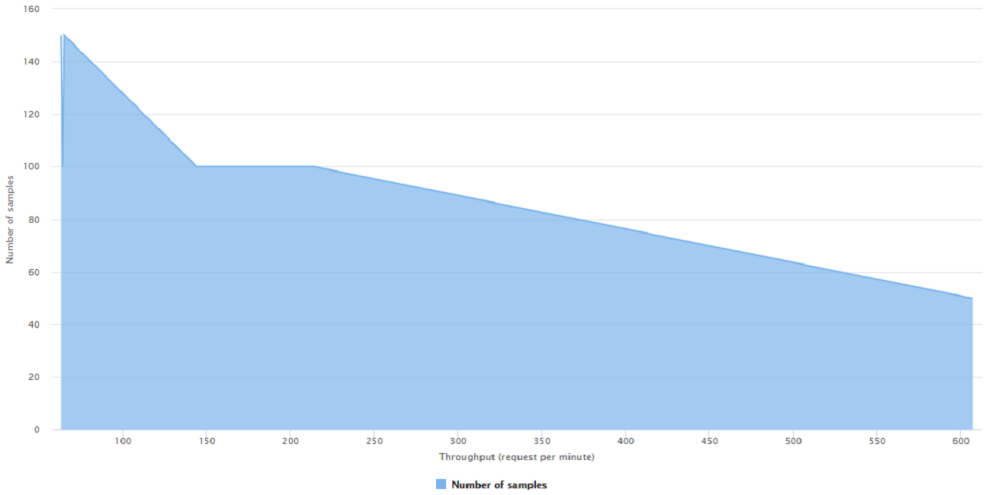

Figura 37 - Desempenho da BD MongoDB em leituras.

Como se pode observar na Figura 37, as escritas num repositório *MongoDB* tendem a diminuir o seu *throughput* proporcionalmente ao aumento de pedidos, isto pode acontecer devido a esta base de dados não lidar com a concorrência, sendo que com 150 *samples*, existem 30 pedidos em simultâneo por segundo, tendo atingido um *throughput* máximo de 607 pedidos por minuto (com 50 utilizadores – 10 por segundo).

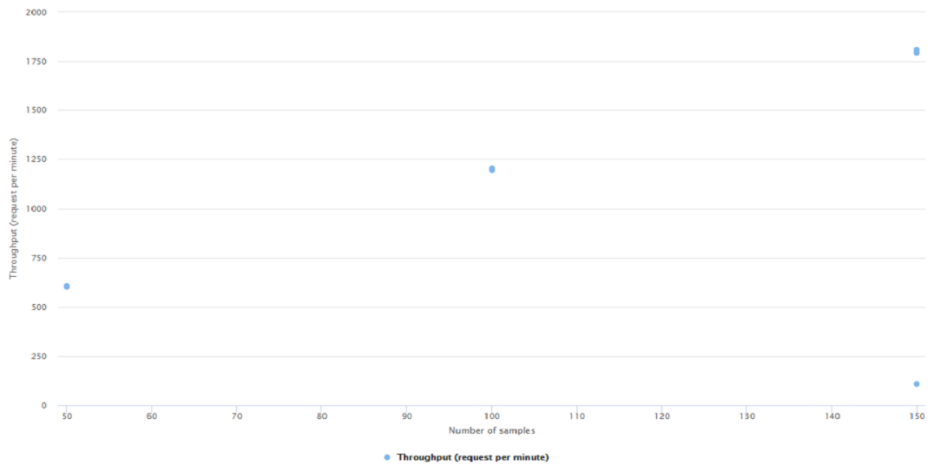

Figura 38 - Desempenho da BD SQL Server em leituras com outlier.

Como se pode observar na Figura 38, em contrapartida, com a base de dados *MongoDB*, a *SQL Server* demonstrou ser muito mais rápida em leituras, atingindo um *throughput* máximo de 1790 pedidos por minuto (com 150 utilizadores – 30 por segundo). Existiu apenas um *outlier* em que esta se demonstrou ter pior desempenho com os números referidos anteriormente.

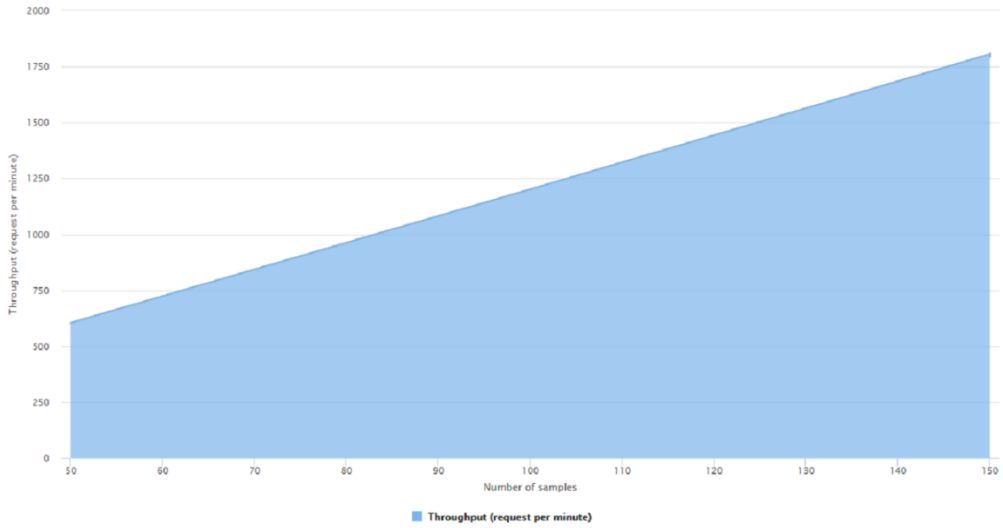

Figura 39 - Desempenho da BD SQL Server em leituras.

Na Figura 39, pode-se observar o gráfico do desempenho da base de dados *SQL Server*, eliminado o *outlier*. Nesta ilustração, a base de dados *SQL Server* tem um comportamento linear, o que corresponde a uma complexidade algorítmica temporal de *O(n)*, que é um excelente desempenho.

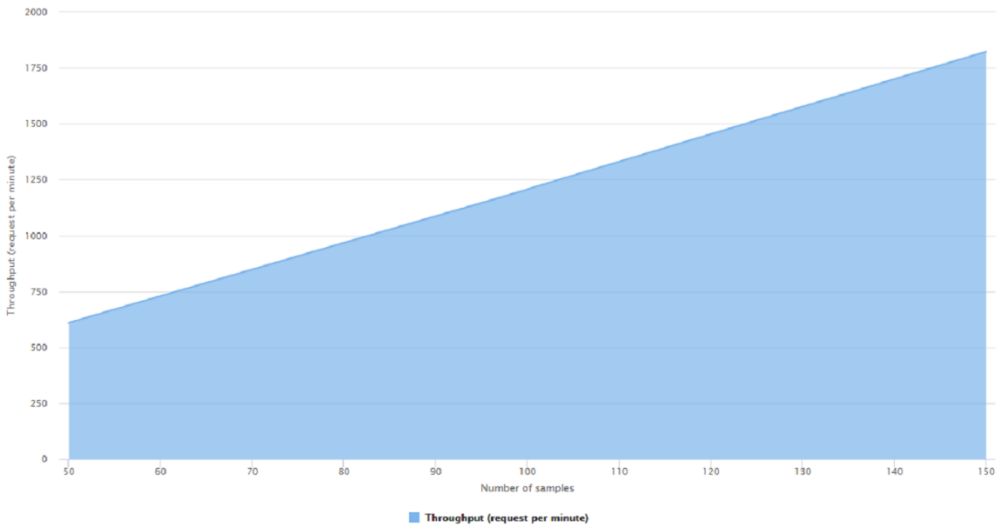

Figura 40 - Desempenho da BD MongoDB em escritas.

Como se pode observar na Figura 40, o tempo de escrita da base de dados *MongoDB* também tem um comportamento linear, podendo ser matematicamente descrito como tendo uma complexidade temporal *O(n)*. De destacar que o *throughput* máximo atingindo foi de 1825 pedidos por minuto, sendo este desempenho atingida com 150 *samples* (150 utilizadores, 30 por segundo).

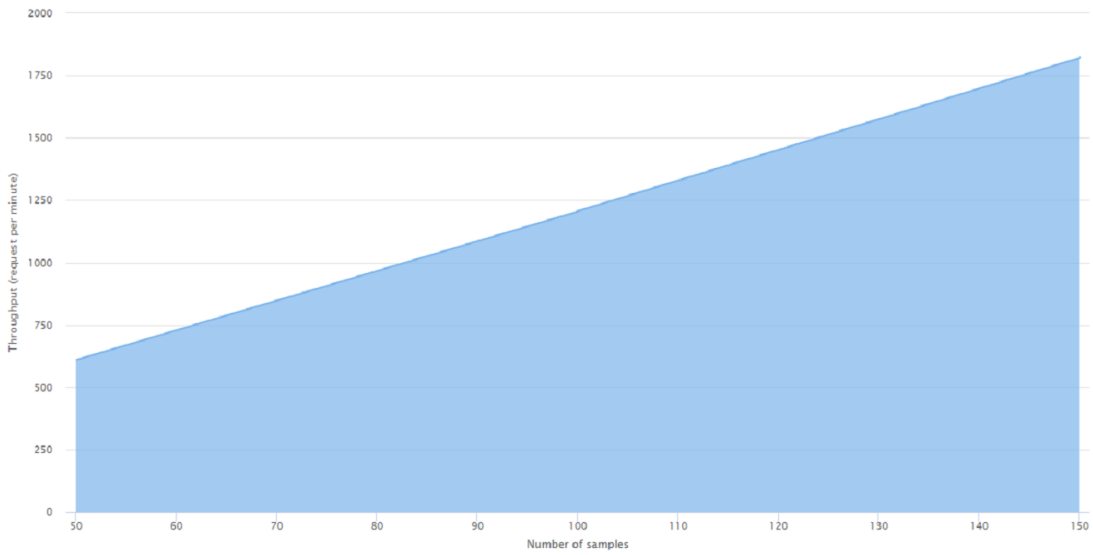

Figura 41 - Desempenho da BD SQL Server em escritas.

Na Figura 41, pode-se observar que a base de dados *SQL Server* também tem um comportamento linear (complexidade temporal *O(n)*) para escrita, tendo atingido um *throughput* máximo de 1820 pedidos por segundo com 150 *samples* (150 utilizadores, 30 por segundo).

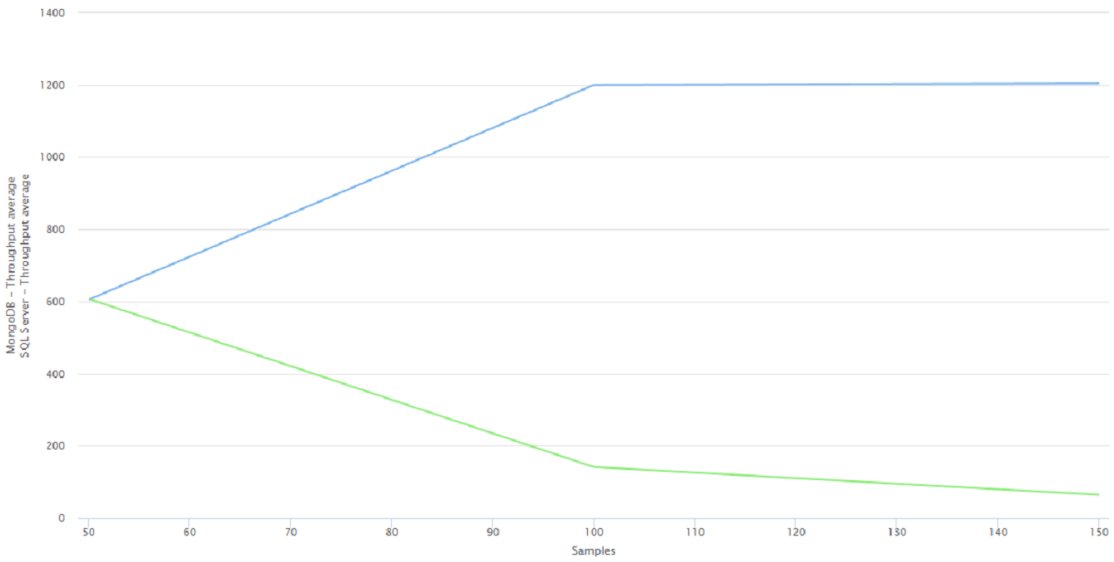

Figura 42 - Comparação entre BD MongoDB e SQL Server em leituras.

Na Figura 42, pode-se observar a comparação de desempenho de leitura entre as duas bases de dados, sendo que ambas demonstram um comportamento linear, proporcional ao número de *samples* (utilizadores).

A base de dados *SQL Server* demonstra ser superior ao nível de leituras, isto também pode-se dever ao fato de a *query* que o ORM estar a produzir ser uma *query* de complexidade baixa. Se esta *query* possui-se o agrupamento (*JOIN*) de várias tabelas, suspeita-se que o resultado seria inferior ao da base de dados *MongoDB*.

Relativamente ao repositório *MongoDB*, observa-se que o comportamento é inversamente proporcional ao número de *samples* (utilizadores), sendo neste caso, inferior em desempenho de leitura ao exibido pelo *SQL Server*.

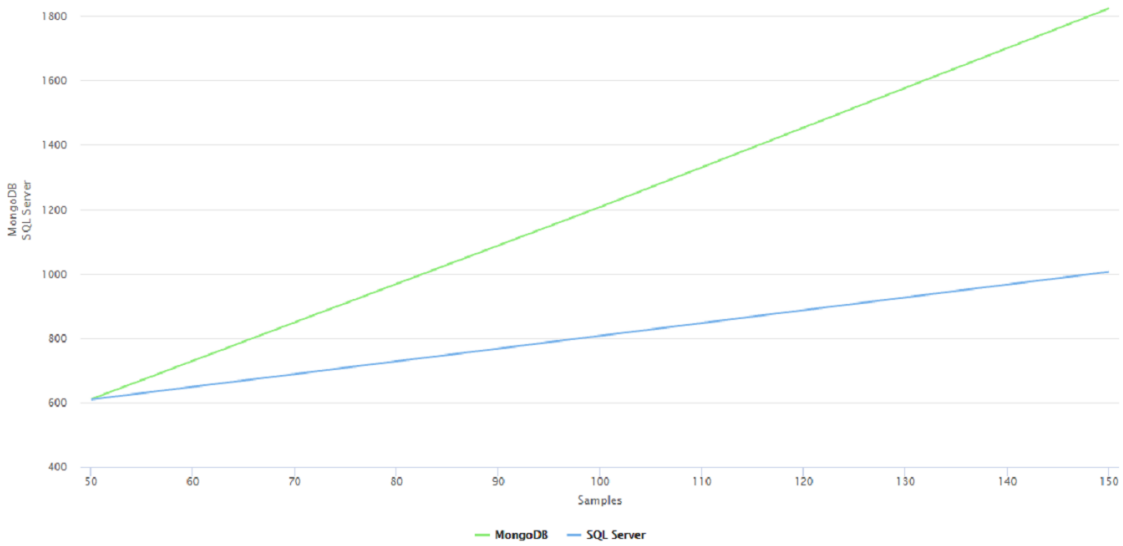

Figura 43 - Comparação entre BD MongoDB e SQL Server em escritas.

Na Figura 43, pode-se observar a diferença de desempenho relativamente à operação de escrita entre as duas bases de dados, sendo que neste caso, *SQL Server* demonstra-se inferior ao *MongoDB*. De referir que o modelo de dados é simplificado comparativamente ao original utilizado no projeto *OEE Web Server*, o que pode implicar que a discrepância seja maior entre estas duas tecnologias, de acordo com o gráfico exibido (i.e., *MongoDB* sendo mais eficiente para escrita que *SQL Server*).

Pelos testes desenvolvidos, e tendo em conta que o modelo de dados era simplificado comparativamente com o modelo de dados original existente no projeto *OEE Web Server*, pode-se concluir que a base de dados *MongoDB* é superior em operações de escrita e inferior em operações de leitura simples. Tal cenário poderia ser diferente se mais testes fossem executados com um modelo de dados mais complexo, que poderia demonstrar que, tal como acontece na literatura [49], *MongoDB* pode-se tornar mais eficiente em operações de leitura devido a não ter de realizar operações de *JOIN* como acontece com repositórios SQL.

Se tal não se verificasse, também se poderia utilizar uma abordagem híbrida que implicaria utilizar as duas tecnologias (um repositório *NoSQL* para escritas em série e um repositório SQL para persistência dos dados) utilizando uma ferramenta de *mirroring* para que ambas se sincronizassem, sendo que esta solução é mais dispendiosa ao nível de licenças e também de custos operacionais.

Os testes neste documento demonstrados são facilmente reproduzíveis, sendo que se espera que os valores possam oscilar consoante o *hardware* utilizado, mas que o comportamento a nível temporal (*e.g.* complexidade de escrita *O(n)* no caso da base de dados *SQL Server*) e espacial sejam similares.

6. Análise crítica

Foram propostas duas perguntas de investigação em âmbito de estado da arte:

- **Q1**: *Que soluções arquiteturais e tecnológicas poderão fomentar uma maior digitalização e otimização de processos de negócio de chão de fábrica?*
- **Q2**: *Como poderá a utilização de protocolos de comunicação standard contribuir para uma maior interoperabilidade entre os principais intervenientes, recursos e dados destes processos?*

Para tentar dar resposta as perguntas propostas, o primeiro passo consistiu na realização de uma revisão sistemática da literatura.

Foi, portanto, desenvolvida uma revisão sistemática que não se limitou à realidade prática que acontece nas fábricas portuguesas, mas que contextualizou a realidade global do estado da arte no que toca à Indústria 4.0. A maior limitação deste estudo foi o de não ter demasiadas referências com artigos que foquem apenas em digitalização de pequenas e médias empresas, sendo que as vantagens e desvantagens dessa digitalização tiveram de ser extrapoladas através de outro tipo de estudos que focavam em empresas de maior dimensão.

Como se pode observar em alguns estudos referidos no estado da arte [50, 51], conseguiu-se a integração homogénea do protocolo OPC-UA com vários tipos de sensores e dados provenientes de sistema de gestão empresarial, como um ERP ou um MES, provando assim a tese dos autores que se focaram neste estudo e também provando que o protocolo OPC-UA, que é um protocolo *standard* na indústria, é indicado para este tipo de aplicação.

As principais finalidades que foram desenvolvidas no âmbito deste trabalho no contexto do projeto *Tooling 4G*, foi a da criação de uma arquitetura de *middleware* que permitisse a integração dos dados provenientes de um sistema de gestão empresarial como um ERP ou um MES com os sistemas de gestão de chão de fábrica, sistemas de aquisição de dados de um chão de fábrica e outros tipos de sistema. Os principais desafios encontrados, prenderam-se com a interoperabilidade que teve de existir, para suportar a utilização de virtualmente qualquer tipo de ERP existente no mercado (sendo que para o efeito foi escolhido um modelo de dados baseado no *standard* ISA-95) que permite facilitar o esforço de transformação de dados para o *middleware*.

Foi realizada uma prova de conceito com o ERP *Primavera* sendo que esta foi testada num servidor de demonstração e aceite pelos principais *stakeholders* do projeto. Seria interessante observar como o *Tooling 4G* se comportaria com uma instância de um MES ao invés de um ERP.

Outro dos principais desafios deste projeto, prendeu-se na interoperabilidade entre sensores, sendo que nesta área, as *interfaces* de comunicação, seja de sensores ou de máquinas que não detenham o protocolo OPC-UA de forma nativa, obrigou ao desenvolvimento de um servidor OPC-UA *ad-hoc* para este efeito, para a inserção de dados provenientes de ordens de fabrico manualmente e também para a comunicação agnóstica com sensores. Para este último caso, foi escolhido um padrão de codificação de dados para envio via rede OPC-UA denominado *SensorML*. *SensorML* define a estrutura e representação dos dados e tipos de dados a serem enviados por sensores.

O sucesso conseguido na integração do *Tooling 4G* com sensores físicos interpola com os resultados obtidos com estudos referidos no estado da arte em que os autores integravam impressoras 3D com sensores *ad-hoc* (e.g. sensores de temperatura), ou placas de circuito de prototipagem como *Arduino Yun*, utilizando o protocolo OPC-UA. Não corroborando a hipótese em teste referentemente à segunda questão científica proposta nesta mesma tese, de que o protocolo OPC-UA iria proporcionar uma boa integração entre serviços. Seria também interessante observar se é possível extrapolar estes resultados com testes reais num ambiente de produção.

Sendo que só se conseguiu até a data de entrega desta tese os testes de integração com uma *Computerized Numerical Control* (CNC) da marca FANUC, seria interessante realizar testes em ambiente real com outros tipos de dispositivos de chão de fábrica, bem como em vários tipos de sensores físicos e averiguar como toda a arquitetura se iria comportar.

A solução *Tooling 4G* encontra-se ainda em finalização noutras componentes não relacionadas com a Informática, e, portanto, ainda não se encontra em produção em várias fábricas de moldes de forma a recolher o *feedback* da sua utilização bem como averiguar quais as suas vantagens ao nível de gastos e eficiência de produção.

No entanto, é previsível que traga vantagens com a visualização de dados de desempenho de uma linha de produção, de uma máquina ou mesmo de uma unidade fabril a tempo real, sendo que com estas é possível ter uma ideia do estado de saúde de uma máquina mesmo que esta não tenha dado um *feedback* de falha, simplesmente pelo facto de se registar uma quebra na produtividade dessa mesma máquina e esta ser observável nos gráficos dos OEE disponíveis no *dashboard* da aplicação *OEE Web Server*.

Com todos estes dados que compõem os gráficos, que são literalmente séries temporais de ordens de fabrico realizadas por máquinas com os seus tempos registados, também é possível retirar conhecimento de todo este volume bruto de dados, se este for trabalhado nesse sentido.

Seria interessante incluir também, neste estudo do estado da arte, alguns *Shop Floor Data Collection Systems* (SFDCS), de forma a estudar também a história destes sistemas e averiguar o tipo de produtos já existentes no mercado. Sendo que o estudo do estado da arte não foi diretamente neste sentido, obteve-se alguma informação relativa a funcionalidades de sistemas de informação desta natureza, mas não se obteve um grande espetro de informação que poderia provavelmente apoiar em certas decisões no decorrer deste projeto, mesmo posteriormente à fase de análise de requisitos.

Num sentido de trabalho futuro, seria interessante observar um trabalho de *Big Data* aplicado ao imenso volume de dados que é produzido pela rede OPC-UA no chão de fábrica, sendo que estes dados são provenientes de sensores e de máquinas, existe contexto e conhecimento que se pode retirar dos dados em formato bruto, se estes forem trabalhados nesse sentido. Para tal, sugere-se uma implementação de uma *pipeline* com a metodologia *Extract – Transform – Load* (ETL) a estes dados, de forma a carregá-los em uma *data warehouse* na *cloud* e utilizar modelos de *data mining* para retirar conhecimento e padrões destes mesmos dados de forma criar um Sistema de Apoio à Decisão e disponibilizar esta informação num *dashboard* online, de forma a ser utilizado pelos decisores de cada empresa que tem esta solução implementada e auxiliar estes em tomadas de decisão.

7. Conclusão

A Indústria 4.0 fomentou novas linhas de investigação no que toca a diferentes áreas multidisciplinares como Engenharia Mecânica, Engenharia Eletrotécnica, Engenharia e Gestão Industrial e Engenharia Informática para citar algumas. A necessidade de uma produção rápida e eficiente, com o mínimo de gastos e desperdícios levou a uma grande mobilização mundial em desenvolver soluções no âmbito do conceito de Indústria 4.0 que permitiram suprir necessidades de produção que outrora não seriam possíveis sem a tecnologia que hoje é empregada nas fábricas.

Este trabalho consistiu num esforço de realizar uma compilação sucinta do estado atual da Indústria 4.0, referindo a breve história das revoluções industriais e os motivadores socioeconómicos que impulsionaram este movimento. Também se constou que existe uma necessidade crescente, não só em Portugal, mas global, de digitalizar os ambientes industriais permitindo o aumento da receita, a redução de custos operacionais e ganhos em eficiência.

Foi realizada uma revisão sistemática da literatura com o objetivo de entender os diferentes graus de digitalização das empresas a nível global – focando não só nas grandes industrias como também nas Pequenas e Médias Empresas (PME), vantagens e desvantagens da implementação das premissas da indústria 4.0, motivadores tecnológicos e áreas de investigação envolvidas no desenvolvimento destas mesmas premissas e também protocolos de comunicação que permitam uma maior fluidez no que toca à digitalização e integração de todo este ecossistema.

Foi descrito um estudo de caso da digitalização de uma fábrica de moldes, sendo que se procedeu à implementação de uma *framework* que permitisse a visualização de dados operacionais e indicadores de desempenho e também uma aplicação para gestão de ordens de fabrico para o chão de fábrica.

Utilizou-se um protocolo de comunicação (i.e., OPC-UA), que permitiu a macro integração em termos arquiteturais de todas estas “peças” que foram construídas separadamente (i.e., todos os serviços da arquitetura *Tooling 4G*).

Garantiu-se requisitos funcionais e não funcionais que tinham sido propostos como a interoperabilidade semântica com a utilização deste protocolo, permitindo a compatibilidade arquitetural desta solução com diferentes sistemas de gestão (i.e., ERP) que já existem no mercado.

Este projeto também possibilitou, graças à utilização do protocolo OPC-UA a ligação agnóstica a vários tipos de sensores, provando o conceito de que é possível a esta solução a comunicação bidirecional com *things* que possam ter diferentes tecnologias, graças a uma das implementações do servidor OPC-UA que foi desenvolvida no âmbito deste projeto.

Sendo que este projeto foi desenvolvido em conjunto com várias empresas que dominam a área de informática de gestão, bem como centros tecnológicos da indústria de moldes, todos os requisitos levantados para o desenvolvimento deste projeto foram feitos numa perspetiva de criação de produto, onde este foi colocado também em pré-produção em servidores de demonstração, posteriormente validado e testado por pessoas com diferentes valências técnicas, criando uma confiança de que esta foi uma boa contribuição quanto à linha de investigação no que toca às tecnologias emergentes da Indústria 4.0.

Sendo a Indústria 4.0 uma área em crescente expansão, com premissas interessantes e bem definidas, todas as contribuições científicas podem ajudar no desenvolvimento desta área, sendo que as suas vantagens são bastante eminentes e não se prendem unicamente à área industrial, mas também ao mercado global.

Com o decréscimo dos custos de produção e o aumento da eficiência e produtividade, é expetável que novos produtos com preços mais acessíveis às massas possam aparecer no mercado.

Referências bibliográficas

[1] "Portugal e a indústria 4.0 - Portugal Têxtil." https://www.portugaltextil.com/portugal-e-a-industria-4-0/ (accessed Feb. 01, 2020).

[2] H. Lasi, H.-G. Kemper, D.-I. T. Feld, and D.-H. M. Hoffmann, "BISE-CATCHWORD The Authors," doi: 10.1007/s12599-014-0334-4.

[3] "What is OPC? - OPC Foundation." https://opcfoundation.org/about/what-is-opc/ (accessed Oct. 31, 2018).

[4] "Plattform Industrie 4.0 - Hintergrund." https://www.plattform-i40.de/PI40/Navigation/DE/Plattform/Hintergrund/hintergrund.html (accessed Nov. 10, 2020).

[5] P. Portugal, "Indústria 4.0." Accessed: Feb. 01, 2020. [Online]. Available: www.pwc.pt/industria40.

[6] D. Moher *et al.*, "Preferred reporting items for systematic review and meta-analysis protocols (PRISMA-P) 2015 statement," 2015. doi: 10.1186/2046-4053-4-1.

[7] D. Marr, *Vision: A Computational Investigation into the Human Representation and ... - David Marr - Google Livros*. 1982.

[8] J. Lee, B. Bagheri, and H. A. Kao, "A Cyber-Physical Systems architecture for Industry 4.0-based manufacturing systems," *Manuf. Lett.*, vol. 3, pp. 18–23, Jan. 2015, doi: 10.1016/j.mfglet.2014.12.001.

[9] F. Pauker, T. Frühwirth, B. Kittl, and W. Kastner, "A Systematic Approach to OPC UA Information Model Design," in *Procedia CIRP*, Jan. 2016, vol. 57, pp. 321–326, doi: 10.1016/j.procir.2016.11.056.

[10] A. N. Lam and O. Haugen, "Implementing OPC-UA services for Industrial Cyber-Physical Systems in Service-Oriented Architecture," *IECON 2019 - 45th Annu. Conf. IEEE Ind. Electron. Soc.*, vol. 1, pp. 5486–5492, 2019, doi: 10.1109/iecon.2019.8926972.

[11] A. Tang, C. Owen, F. Biocca, and W. Mou, “Comparative Effectiveness of Augmented Reality in Object Assembly.”

[12] E. Ras, F. Wild, C. Stahl, and A. Baudet, “Bridging the skills gap of workers in industry 4.0 by human performance augmentation tools - Challenges and roadmap,” in *ACM International Conference Proceeding Series*, Jun. 2017, vol. Part F128530, pp. 428–432, doi: 10.1145/3056540.3076192.

[13] F. W. Baumann, M. Falkenthal, S. Hudert, U. Odefey, and M. Zimmermann, “Cyber-physical System Control via Industrial Protocol OPC UA Institute of Architecture of Application Systems Cyber-physical System Control via Industrial Protocol OPC UA,” 2017, [Online]. Available: http://www.iaria.org/conferences2017/ADVCOMP17.html.

[14] National Instruments, “Introdução ao Modbus,” 2014. http://www.ni.com/white-paper/7675/pt/.

[15] A. Joe Turner, U. Editorial Board, B. Meyer, E. Zurich, K. Rannenberg, and M. A. Bramer, “IFIP Advances in Information and Communication Technology 311 Editor-in-Chief Security and Privacy Protection in Information Processing Systems.”

[16] E. Tovar and F. Vasques, “Real-time fieldbus communications using Profibus networks,” *IEEE Trans. Ind. Electron.*, vol. 46, no. 6, pp. 1241–1251, 1999, doi: 10.1109/41.808018.

[17] “EtherNet/IP: Industrial Protocol White Paper.”

[18] B. Galloway and G. P. Hancke, “Introduction to industrial control networks,” *IEEE Commun. Surv. Tutorials*, vol. 15, no. 2, pp. 860–880, 2013, doi: 10.1109/SURV.2012.071812.00124.

[19] M. Ghobakhloo, “The future of manufacturing industry: a strategic roadmap toward Industry 4.0,” *J. Manuf. Technol. Manag.*, vol. 29, no. 6, pp. 910–936, Oct. 2018, doi: 10.1108/JMTM-02-2018-0057.

[20] M. Schleipen, S. S. Gilani, T. Bischoff, and J. Pfrommer, “OPC UA & Industrie 4.0 - Enabling Technology with High Diversity and Variability,” *Procedia CIRP*, vol. 57, pp. 315–320, 2016, doi: 10.1016/j.procir.2016.11.055.

[21] J. Pfrommer and F. Palm, “RESTful Industrial Communication With OPC-UA,” vol. 12, no. 5, pp. 1832–1841, 2016.

[22] M. Muller, E. Wings, and L. Bergmann, “Developing open source cyber-physical systems for service-oriented architectures using OPC UA,” *Proc. - 2017 IEEE 15th Int. Conf. Ind. Informatics, INDIN 2017*, pp. 83–88, 2017, doi: 10.1109/INDIN.2017.8104751.

[23] H. Derhamy, J. Ronnholm, J. Delsing, J. Eliasson, and J. Van Deventer, “Protocol interoperability of OPC UA in service oriented architectures,” *Proc. - 2017 IEEE 15th Int. Conf. Ind. Informatics, INDIN 2017*, pp. 44–50, 2017, doi: 10.1109/INDIN.2017.8104744.

[24] F. Xinyang, S. Jianjing, and F. Ying, “REST: An alternative to RPC for web services architecture,” in *2009 1st International Conference on Future Information Networks, ICFIN 2009*, 2009, pp. 7–10, doi: 10.1109/ICFIN.2009.5339611.

[25] “P of EAA: Data Transfer Object.” https://martinfowler.com/eaaCatalog/dataTransferObject.html (accessed Dec. 07, 2018).

[26] International Society of Automation, “ISA 95 - About.” https://isa-95.com/#about (accessed Dec. 07, 2018).

[27] M. Fowler, “P of EAA: Remote Facade.” https://martinfowler.com/eaaCatalog/remoteFacade.html (accessed Dec. 07, 2018).

[28] “AutoMapper.” https://automapper.org/ (accessed Dec. 07, 2018).

[29] S. Nakajima, “Introduction to TPM: Total Productive Maintenance - Seiichi Nakajima - Google Livros,” 1988. https://books.google.pt/books?id=XKc28H3JeUUC&q=Nakajima,+S.+(1988),+Introduction+to+Total+Productive+Maintenance+(TPM),+Productivity+Press,+Portland&dq=Nakajima,+S.+(1988),+Introduction+to+Total+Productive+Maintenance+(TPM),+Productivity+Press,+Portland& (accessed Jul. 25, 2020).

[30] P. Muchiri and L. Pintelon, “Performance measurement using overall equipment effectiveness (OEE): literature review and practical application discussion,” *Int. J. Prod. Res.*, vol. 46, no. 13, pp. 3517–3535, Jul. 2008, doi: 10.1080/00207540601142645.

[31] “Highcharts for C#.” https://dotnet.highcharts.com/ (accessed Aug. 24, 2020).

[32] “Chart.js | Open source HTML5 Charts.” https://www.chartjs.org/ (accessed Aug. 24, 2020).

[33] “Entity Framework documentation | Microsoft Docs.” https://docs.microsoft.com/en-us/ef/ (accessed Aug. 24, 2020).

[34] “The Basics of Information Security: Understanding the Fundamentals of ... - Jason Andress - Google Livros.” https://books.google.pt/books?id=E3jTrBwpWPoC&pg=PA6&dq=cia+triad&hl=pt-PT&sa=X&ved=2ahUKEwjBkJLC0bbrAhVRDWMBHWeYByoQ6AEwAHoECAMQAg#v=onepage&q=cia triad&f=false (accessed Aug. 25, 2020).

[35] “Computing: A Historical and Technical Perspective - Yoshihide Igarashi, Tom Altman, Mariko Funada, Barbara Kamiyama - Google Livros.” https://books.google.pt/books?id=58ySAwAAQBAJ&pg=PA55&dq=De+Bello+Gallico+caesar+cypher&hl=pt-PT&sa=X&ved=2ahUKEwj87OGK0LbrAhVSJhoKHVbBDnUQ6AEwAHoECAEQAg#v=onepage&q=De Bello Gallico caesar cypher&f=false (accessed Aug. 25, 2020).

[36] “Unified Architecture - OPC Foundation.” https://opcfoundation.org/about/opc-technologies/opc-ua/ (accessed Feb. 01, 2020).

[37] E. Kucera, O. Haffner, P. Drahos, and S. Kozak, “(PDF) Emerging Technologies for Industry 4.0: OPC Unified Architecture and Virtual / Mixed Reality,” *Conference: AIFICT 2018 - Applied Informatics in Future ICTAt: Bratislava,* Apr. 2018. https://www.researchgate.net/publication/324952880_Emerging_Technologies_for_Industry_40_OPC_Unified_Architecture_and_Virtual_Mixed_Reality (accessed Sep. 29, 2020).

[38] OPC Foundation, “OPC-UA Standard.” https://opcfoundation.org/developer-tools/specifications-unified-architecture (accessed Aug. 27, 2020).

[39] OPC Foundation, “Unified Architecture - OPC Foundation.” https://opcfoundation.org/about/opc-technologies/opc-ua/ (accessed Aug. 26, 2020).

[40] International Society of Automation, “ANSI/ISA-95.” https://isa-95.com/ (accessed Sep. 01, 2020).

[41] “FANUC OPC Server - Fanuc.” https://www.fanuc.eu/pt/pt/cnc/connectivity/opc-server (accessed Oct. 12, 2020).

[42] Google, “Flutter - Beautiful native apps in record time.” https://flutter.dev/?gclid=Cj0KCQjwhb36BRCfARIsAKcXh6E312iBLfONMoYlcxP-tjmjhk1OjdB1Mgqo2uOv9dIzwkoFuJkTFNQaAjAuEALw_wcB&gclsrc=aw.ds (accessed Sep. 02, 2020).

[43] Microsoft, “Xamarin | Open-source mobile app platform for .NET.” https://dotnet.microsoft.com/apps/xamarin (accessed Sep. 02, 2020).

[44] Internet Engineering Task Force (IETF), “RFC 6455 - The WebSocket Protocol,” Dec. 2011. https://tools.ietf.org/html/rfc6455 (accessed Sep. 03, 2020).

[45] SonarSource, “SonarQube.” https://www.sonarqube.org/ (accessed Sep. 15, 2020).

[46] C. Poole and R. Prouse, “NUnit.org.” https://nunit.org/ (accessed Sep. 24, 2020).

[47] Apache Software Foundation, “Apache JMeter - Apache JMeter™.” https://jmeter.apache.org/ (accessed Sep. 25, 2020).

[48] C.-O. Truică, F. Rădulescu, A. Boicea, and I. Bucur, “Performance evaluation for CRUD operations in asynchronously replicated document oriented database,” Jun. 2018, doi: 10.1109/CSCS.2015.32.

[49] Y. Li and S. Manoharan, “A performance comparison of SQL and NoSQL databases,” in *2013 IEEE Pacific Rim Conference on Communications, Computers and Signal Processing (PACRIM)*, Aug. 2013, pp. 15–19, doi: 10.1109/PACRIM.2013.6625441.

[50] F. Longo, L. Nicoletti, and A. Padovano, “Ubiquitous knowledge empowers the Smart Factory: The impacts of a Service-oriented Digital Twin on enterprises’ performance,” *Annu. Rev. Control*, vol. 47, pp. 221–236, Jan. 2019, doi: 10.1016/j.arcontrol.2019.01.001.

[51] S. Cavalieri, D. Di Stefano, M. G. Salafia, and M. S. Scroppo, “A web-based platform for OPC UA integration in IIoT environment,” *IEEE Int. Conf. Emerg. Technol. Fact. Autom. ETFA*, pp. 1–6, 2017, doi: 10.1109/ETFA.2017.8247713.