Universidade de Aveiro Departamento de Engenharia Mecânica
2021

Ângelo Francisco Alves Pereira

Desenvolvimento de Software Para Conectar um CNC ao Nexeed MES

Development of Software to Connect a CNC to Nexeed MES

Universidade de Aveiro Departamento de Engenharia Mecânica
2021

Ângelo Francisco Alves Pereira

Desenvolvimento de Software Para Conectar um CNC ao Nexeed MES

Development of Software to Connect a CNC to Nexeed MES

Relatório de Projeto apresentada à Universidade de Aveiro para cumprimento dos requisitos necessários à obtenção do grau de Mestrado em Engenharia Mecânica, realizada sob orientação científica de José Paulo Oliveira Santos, Professor Auxiliar do Departamento de Engenharia Mecânica da Universidade de Aveiro e de Engº Duarte Almeida, Supervisor na BOSCH Termotecnologia.

Este trabalho teve o apoio financeiro dos projetos UIDB/00481/2020 e UIDP/00481/2020 - FCT - Fundação para Ciência e Tecnologia; e CENTRO-01-0145-FEDER-022083- Programa Operacional Regional do Centro (Centro2020), no âmbito do Acordo de Parceria Portugal 2020, através do Fundo Europeu de Desenvolvimento Regional.

O júri / The jury

Presidente / President	**Prof. Doutor António Manuel Godinho Completo** Professor Associado C/ Agregação da Universidade de Aveiro
Vogais / Committee	**Prof. Doutor Paulo Bacelar Reis Pedreiras** Professor Auxiliar da Universidade de Aveiro
	Prof. Doutor José Paulo Oliveira Santos Professor Auxiliar da Universidade de Aveiro (orientador)

Agradecimentos / Acknowledgements

Agradeço Prof. Doutor José Paulo Santos, por todo o acompanhamento, orientação e disponibilidade.
Ao Eng. Duarte Almeida, por todo o seu acompanhamento, profissionalismo e pela disponibilidade incansável que demonstrou durante esta minha etapa académica.
À Bosch Termotecnologia, pelo seu acolhimento e integração disponibilizando todos os meios necessários à realização deste trabalho.
À minha família, Pai Mãe e Irmão, por todo o apoio, dedicação, disponibilidade e ajuda.
Ao meu amigo Diogo Costa pelo seu apoio e amizade.
A todos os meus amigos e colegas que me acompanharam e apoiaram.

Palavras-chave CNC, MES, Monitorização, OPC, Aquisição de dados

Resumo

A melhoria da qualidade e do desempenho dos produtos e processos produtivos constituem uma busca constante para qualquer empresa ou empresário, a fim de se manterem competitivos no mercado. Consequentemente, os fabricantes dos dias de hoje escolhem equipamentos CNC (Controlo Numérico por Computador) em detrimento de tecnologias tradicionais, permitindo a realização de uma maior variedade de tarefas com maior precisão, repetibilidade e com um controlo mais automatizado.

Sistemas de Execução de Manufatura (MES) são fundamentais na gestão das atividades de produção, para isso estabelecendo uma ligação entre o planeamento e o chão de fábrica. Estes sistemas estão cada vez mais difundidos pela indústria, sendo o seu foco principal o processamento de informações em tempo real, de tal forma que culminem numa otimização do processo produtivo.

Este trabalho foi desenvolvido em cooperação com a *BOSCH Termotecnologia*, tendo como objetivo principal o encaminhamento de informações em tempo real, com origem num equipamento CNC, *Chiron FZ 15 K W*, presente na instalação fabril, para o sistema MES utilizado pela empresa, que é o *Nexeed MES*.

No âmbito do presente trabalho, foram desenvolvidos programas que permitem a implementação desta partilha de informação, fazendo uso de tecnologias OPC (Open Platform Communications) no processo de aquisição de informações provenientes da CNC e de telegramas XML (Extensible Markup Language) para o fornecimento das mesmas ao sistema MES.

Os resultados obtidos mostram que o sistema desenvolvido poderá desempenhar um papel fundamental no aumento da produtividade e do nível monitorização de equipamentos CNC em particular no equipamento *legacy* sem capacidades de *networking*. A arquitetura desenvolvida pode ser facilmente introduzida num ambiente industrial com a finalidade de monitorizar equipamentos deste tipo.

Keywords CNC, MES, Monitoring, OPC, Data aquisition

Abstract

The continuous improvement of quality and performance of products and productive systems is a requirement for any company or entrepreneur that wishes to remain competitive in the market. Consequently, today's manufacturers choose CNC (Computer Numerical Control) equipment over traditional technologies, allowing the accomplishment of a greater variety of tasks with higher precision, repeatability and with more automated control. Manufacturing Execution Systems (MES) are a key element in the management of production activities, establishing a link between planning stages and the shop floor. These systems are becoming more and more widespread in industry, and their focus is processing information in real time in a way that optimizes every step of the production process.
This work was developed in cooperation with *BOSCH Termotecnologia*, and the main objective was forwarding information in real time, from a CNC, *Chiron FZ 15 KW* equipment, located in the factory, to the MES system used by the company (*Nexeed MES*).
Within the scope of this work, programs were developed that allows the implementation of this type of information sharing, using OPC (Open Platform Communications) technologies in the acquisition of information from the CNC, and XML (Extensible Markup Language) telegrams to supply them to the MES system.
The obtained results show that the developed system can play a fundamental role in increasing productivity and monitoring level of CNC equipment, in particular on the textit legacy device without textit networking features. The developed architecture can be easily introduced in an industrial environment with the purpose of monitoring equipment of this type.

Índice

Lista de Tabelas

Lista de Figuras

Lista de Código Fonte

Acrónimos

AGV Automated Guided Vehicle

APT Automatically Programmed Tool

CAD Computer Aided Design

CAM Computer Aided Manufacturing

CNC Controlo Numérico por Computador

COM Component Object Model

DB Data Block

DCE Distributed Computing Environment

DCOM Distributed Component Object Model

DDE Dynamic Data Exchange

DDL Direct Data Link

DHCP Dynamic Host Configuration Protocol

DLL Dynamic Link Library

DNC Direct Numerical Control

DPR Dual-Port Ram

EPROM Flash Erasable Programmable Read Only Memory

ERP Enterprise Resource Planning

FB Function Block

FC Function Call

HMI Human-Machine Interface

IOT Internet of Things

IP Internet Protocol

KPI Key Performance Indicator

MCP Machine Control Panel

MDI Manual Data Input

MES Manufacturing Execution System

MMC Man Machine Communication

MPI Multi-Point Interface

MRP Material Requirement Planing

NC Numeric Control

NCK Numeric Control Kernel

NCU Numeric Control Unit

NIC Network Interface Card

OB Organization Block

OEE Overall Equipment Effectiveness

OLE Object Linking and Embedding

OPC Open Platform Communications

PCU Computer Unit

PI Process Integration

PLC Programmable Logic Controller

RAM Random Access Memory

RPC Remote Procedure Call

SCADA Supervisory Control And Data Acquisition

SDK Software Development Kit

SPC Statistical Process Control

SPL Safe Programmable Logic

TCP/IP Transmission Control Protocol - Internet Protocol

XML Extensible Markup Language

XSD XML Schema Definition

Glossário

DDE Dynamic Data Exchange - Tecnologia para a comunicação e manipulação de objetos entre programas implementados nos primeiros sistemas operativos *Microsoft Windows*. Devido à sua vulnerabilidade a *malwares* foi substituído pela tecnologia OLE (Object Linking and Embedding). O último sistema operativo a suportar o DDE foi o *Windows XP* [1].

OLE Object Linking and Embedding - Tecnologia desenvolvida pela *Microsoft* em 1990 com o objetivo de integrar diferentes aplicações *Windows*. Esta tecnologia é uma evolução da tecnologia DDE que estava limitada na quantidade de informação transmitida entre duas aplicações [2].

DCE/RPC Sistema de chamadas de procedimento remoto desenvolvido para o DCE (Distributed Computing Environment). Este sistema permite a escrita de programas para funcionarem em computadores diferentes como se estivessem no mesmo sem haver necessidade de acautelar o sistema de comunicação da rede. É baseado na arquitetura cliente-servidor, as comunicações podem ser encapsuladas em mensagens *TCP, UDP, SMB e SMB2* [3].

Firewall Sistema de segurança que monitoriza e controla o tráfico em redes de computadores com base em regras pré-determinas [4].

IPC Pipes Sistema de comunicação entre processos baseados em secções de memória partilhada por vários processos. A aplicação que cria o *pipe* é o servidor e o processo que se conecta é o cliente [5].

Capítulo 1

Introdução

Nos dias que correm, assistimos cada vez mais à implementação de uma economia à escala global, na qual o fluxo de informação e produtos é feito de maneira instantânea, encontrando-se na maioria dos casos à distancia de um simples *click*. Com o elevado desenvolvimento tecnológico nos últimos anos, o mundo ficou de certa forma mais pequeno, sendo que hoje é mais fácil do que nunca aceder a produtos oriundos de países distantes. Perante esta situação, a competitividade empresarial intensificou-se, uma vez que as empresas não competem só com concorrentes geograficamente próximos, mas também com empresas espalhadas por todo o mundo.

Para que uma empresa se afirme no mercado em que concorre, é fundamental que cumpra requisitos como a elevada eficiência, baixo custo, elevada flexibilidade, baixo consumo energético e baixo desperdício. De modo a satisfazer as exigências dos mercados atuais torna-se indispensável a implementação de políticas que visem a melhoria contínua. O processo de melhoria contínua necessita de grandes quantidades de informação que depois de analisada e processada, permite a identificação de *bottlenecks* e fatores limitantes de eficiência.

Centros de maquinagem equipados com comandos numéricos por computador são a base da indústria moderna de transformação, logo a sua monitorização é fundamental para o processo de otimização de produção. Este tipo de equipamentos permite não só um aumento de precisão e de repetibilidade, como também um aumento de versatilidade, cadência e um controlo mais automatizado. Devido à irrefutabilidade das suas vantagens este equipamentos CNC substituíram tecnologias mais tradicionais e estão largamente difundidos pela indústria, podendo ser encontrados em qualquer empresa, desde pequenas indústrias locais, até a gigantes multinacionais.

Com o aumento de dados e informações recolhidas no chão de fábrica surgiu a necessidade de um sistema que seja capaz de adquirir, analisar, processar e disponibilizar dados de todos os equipamentos. Para dar resposta a este problema, entram em campo os Sistemas de Execução da Manufatura que, com os seus variados módulos permitem uma aquisição centralizada de informação e dados referentes a produtos e a processos produtivos, disponibilizando de forma imediata e em tempo real o cálculo de vários OEE (Overall Equipment Effectiveness) e KPIs (Key Performance Indicators). Deste modo, torna-se possível a monitorização de indicadores de eficiência de todos os processos do chão de fábrica, desde o planeamento até à produção, contribuindo de forma rápida e direta para o aumento da eficiência e diminuição de custos.

A empresa *BOSCH Termotecnologia* possui nas suas instalações equipamentos CNC

com vários anos de produção. Apesar do sistema MES implementado nesta empresa, o *Nexeed* MES [6], possuir diversos módulos de comunicação com equipamentos de produção, não disponibiliza, porém, nenhuma forma de monitorização de equipamentos CNC antigos. Como objeto de estudo deste trabalho, utiliza-se o equipamento CNC *Chiron FZ 15K W* produzido nos inícios do século 21, a partir do qual se pretende desenvolver um sistema que permita a sua monitorização em tempo real assente no envio de dados para o *Nexeed MES*.

1.1 Objetivos

O presente trabalho tem por objetivo desenvolver uma solução que permita a aquisição de dados de um centro de maquinagem equipado com um comando numérico *Siemens* (*Chiron FZ 15K W*), que se encontra a laborar na empresa *Bosch Termotecnologia*. A solução a desenvolver deve ser capaz de comunicar os parâmetros recolhidos ao sistema MES implementado na empresa.

Os dados a recolher necessitam de ser selecionados de tal forma que:

- Caracterizem e quantifiquem as peças produzidas.
- Contabilizem o desgaste das ferramentas em uso.
- Permitam alertar para a ocorrência de alarmes.
- Permitam identificar *bottlenecks* na produção.
- Possibilitem o cálculo de OEEs (Overall Equipment Effectiveness) e KPIs (Key Performance Indicators).

Os dados deverão ser recolhidos o mais instantaneamente possível. Alarmes de elevada severidade necessitam de ser comunicado ao MES prontamente. Por outro lados os parâmetros de funcionamento deverão ser transmitidos no final de um ciclo de maquinagem.

O sistema deverá fornecer dados fiáveis, ser de fácil implementação, facilmente adaptável a novos requisitos e funcionalidades e possuir um elevado grau de robustez dado que será implementado num ambiente industrial.

1.2 Motivação

Um estudo publicado [7] na mais prestigiada revista dedicada à maquinação CNC dos Estado Unidos da América, a *Modern machine shop*, afirma que para um universo de 350 empresas do setor, cerca de 75 % tinham planos para a instalação de um sistema de monitorização dos seus equipamentos CNC, à data de realização deste estudo, 2017. Isto demonstra que os profissionais desta ramo estão cientes das grandes vantagens da monitorização dos seus equipamentos.

Com o tremendo consumo de combustíveis fosseis, é imperativo que a sociedade atual encontre formas de diminuir o consumo de energia e consequentemente baixar as emissões de poluentes. Perante esta premissa, a grande motivação deste trabalho é o desenvolvimento de um sistema que permita o aumento da eficiência e a diminuição das perdas e desperdício num processo de maquinação com recurso a um equipamento CNC.

Com um aumento da eficiência do processo serão produzidas mais peças para o mesmo gasto energético, o que por sua vez acarreta uma diminuição no consumo de energia e um aumento dos ganhos da empresa. Este excedente de ganhos poderá ser direcionado para o aumento da eficiência de outros setores produtivos.

1.3 Organização do documento

O presente projeto encontra-se divido em seis etapas, distribuídas da seguinte forma:

- Bosch Termotecnologia/Equipamento - Apresentação da empresa e caracterização do equipamento.
- Revisão do Estado da Arte em Monitorização CNC - Estudo da interface de comunicação do equipamento, estudo do sistema MES em funcionamento na empresa e levantamento de soluções existentes.
- Solução Conceptual - Apresentação do conceito da solução, apresentação e descrição da composição e funcionamento do *Nexeed* MES e uma análise detalhada dos constituintes do equipamento CNC
- Solução Proposta - Apresentação da arquitetura proposta e implementada
- Análise de Resultados - Análise do desempenho do sistema desenvolvido
- Conclusões - Conclusões finais, trabalhos futuros e dificuldades na execução.

Capítulo 2

Bosch Termotecnologia/Equipamento

Este capítulo apresenta o grupo *Bosch*, discutindo a presença da *Bosch* em Portugal e, em maior detalhe a empresa *Bosch* Termotecnologia. É também apresentado de forma breve o equipamento no qual se debruçou o trabalho.

2.1 Grupo *Bosch*

O grupo *Robert Bosch GmbH* é uma empresa multinacional alemã de engenharia e eletrónica fundada em 1886 por *Robert Bosch*, atualmente sediada em Gerlingen, Alemanha. As atividades deste grupo dividem-se em quatro principais setores: tecnologia de energia e edifícios (segurança, domótica e termotecnologia), tecnologia industrial (sistemas de acionamento e controlo), bens de consumo (eletrodomésticos e ferramentas elétricas) e soluções de mobilidade (inclui sistemas de injeção, sistemas de segurança e sistemas de controlo) [8]. Na tabela 2.1, encontra-se representada a distribuição do volume total de vendas do grupo segundo os dados referentes ao ano de 2018.

Tabela 2.1: Distribuição do volume de vendas pelos diferentes setores [8].

Setor	Contribuição
Soluções de mobilidade	61%
Bens de consumo	23%
Tecnologia industrial	9%
Tecnologia de energia e edifícios	7%

O grupo *Bosch* é detido em 92% pela instituição de caridade sem fins lucrativos *Robert Bosch Stiftung GmbH*, que tem a seu cargo atividades filantrópicas e sociais. Esta organização e estrutura de propriedade permite que o grupo reinvista grande parte dos seus lucros em na construção do seu futuro e no crescimento sustentável. Cerca de 9% dos lucros são direcionados para a investigação e desenvolvimento, sendo o dobro da média praticada pela indústria, culminando num maior avanço tecnológico dos produtos

produzidos que se traduz numa liderança de mercado na maiorias das áreas de ação do grupo [9].

Em 2019 este grupo obteve 77.9 mil milhões de euros em vendas, contando atualmente com 410 mil colaboradores, dos quais cerca de 70 mil se dedicam a investigação e desenvolvimento. Este universo de colaboradores encontra-se espalhados por 440 subsidiárias presentes em 60 países [8].

2.2 *Bosch* Portugal

O grupo *Bosch* opera em Portugal desde 1911, com operações no setor da mobilidade, energia e edifícios, tabela 2.2. Segundo os dados de 2018, o grupo emprega 5503 colaboradores e gerou um volume de vendas de 1.7 mil milhões de euros dos quais 95% são referentes a exportações.

Tabela 2.2: *Bosch* em Portugal [10].

Localização	Setor
Aveiro	*Bosch* Termotecnologia
Braga	*Bosch* Car Multimedia
Lisboa	Sede(vendas, *markting* e contabilidade)
Ovar	*Bosch Security Systems*

Apesar da sua pequena dimensão, quando comparada com o grupo do qual faz parte, a *Bosch* Portugal tem um enorme impacto na economia do país, tanto pelo número de cidadãos que emprega como pelo volume de exportação que efetua. Esta empresa conta com uma presença consolidada em Portugal estando, entre as 20 maiores empresas do país [10].

2.3 *Bosch* Termotecnologia

Bosch Termotecnologia S.A é uma divisão do grupo *Robert Bosch GmbH* que se dedica à gestão da unidade de negócios de água quente residencial. Esta empresa está sediada em Aveiro, tendo iniciado laboração em 1988 após a aquisição da empresa portuguesa Vulcano Termodomésticos.

Conta com 164 mil metros quadrados de instalações, figura 2.1, nas quais operam cerca de 1300 colaboradores no desenvolvimento e produção de esquentadores, caldeiras e bombas de calor para várias marcas do grupo *Bosch*. Com uma produção anual de 1.1 milhões de equipamentos, representando 170 milhões em vendas, assume-se como um dos maiores dinamizadores da economia da região de Aveiro [11].

Figura 2.1: Vista aérea da Bosch Termotecnologia.

2.4 Equipamento

Neste secção será feita uma apresentação do equipamento alvo de estudo e do processo de maquinagem que ocorre no mesmo.

2.4.1 *Chiron FZ 15K*

O equipamento no qual o presente projeto se debruça é uma CNC de fresagem vertical de 3 eixos da marca *Chiron*, modelo *FZ 15K W magnum high speed*, figura 2.2. É controlada por um *Siemens Sinumerik 840D* equipado com um HMI (Human-Machine Interface) *PCU 50* [12]. É deste equipamento que se pretende retirar parâmetros e dados durante o seu funcionamento.

A empresa *CHIRON*, sediada na Alemanha, teve a sua origem em 1921 com a produção de equipamento cirúrgico, evoluindo para o fabrico de componentes de guerra e compressores. Atualmente, dedica-se inteiramente a construção de equipamentos CNC com elevada velocidade de rotação da árvore [13].

O facto de possuir uma mesa de trabalho rotativa é uma característica diferenciadora e atípica deste modelo. Esta característica permite ao trabalhador fixar peças a serem maquinadas e retirar peças acabadas, ao mesmo tempo que a máquina executa o ciclo de maquinação em novas peças.

Figura 2.2: Chiron FZ 15K.

Características da *Chiron FZ 15 K*

Em seguida, são apresentadas as principais características do equipamento:

- **Número de eixos:** 4
- **Eixo X:** 550 mm
- **Eixo Y:** 400 mm
- **Eixo Z:** 425 mm
- **Eixo A:** $+180^{o}$ - 180^{o}
- **Largura da mesa:** 400 mm
- **Comprimento da mesa:** 630 mm
- **Potência máxima:** 58 kW
- **Velocidade máxima da árvore:** 20000 rpm
- **Velocidade de avanço máxima** 12 m/min
- **Aceleração máxima eixo X,Y,Z:** 12 m/s^2
- **Número de ferramentas:** 48
- **Peso máximo da ferramenta:** 10,0 kg
- **Diâmetro máximo da ferramenta:** 175 mm
- **Comprimento máximo da ferramenta:** 300 m
- **Peso:** 7220 Kg
- **Ano de fabrico:** 2006
- **Controlador:** *Siemens Sinumerik 840D* equipado com *HMI PCU 50*

Tratando-se de uma CNC de fresagem vertical, esta maquina é constituída por:

- **Estrutura principal:** Responsável por albergar todos os outros componentes. Por norma é um componente de elevada massa com o intuito de diminuir a deflexão e a propagação de vibrações
- **Árvore:** Proporciona a fixação e rotação da ferramenta. O movimento nos eixos X, Y e Z dá-se neste componente.
- **Mesa:** Base para a fixação dos elementos a maquinar, responsável pelo movimento do eixo A (rotação em X).
- **Armazém de ferramentas:** Componente onde são armazenadas as ferramentas.
- ***PLC*:** Responsável pela gestão das entradas e saídas digitais e analógicas.
- ***NCK*:** Responsável pela gestão do funcionamento.

- **Sistema Motriz:** Sistema responsável pelo movimento da *CNC* definidos pelo do *NCK*.
- **Sistema de medida:** Completa o ciclo fechado do controlo de movimentos e velocidades.
- **Monitor:** Responsável pela transmissão de toda a informação relativa ao funcionamento.

Como referido anteriormente, esta CNC possui uma plataforma giratória que agrega duas mesas de trabalho. Num dado momento, uma das mesas encontra-se virada para o operador, enquanto a outra está orientada com a árvore da máquina em regime de maquinação. Quando se completa um ciclo de maquinagem, a plataforma gira 180^{o} a fim de inverter a posição das mesas.

Cada mesa tem um eixo rotacional na direção X, eixo A, que permite a execução de processos de maquinação (furação, roscagem, fresagem) com diferentes ângulos. É importante ainda referir que apesar de possuir 4 eixos, não é possível a interpolação de todos, mas sim apenas dos eixos X, Y e Z, o eixo A tem por objetivo a rotação da peça em relação à árvore.

As peças a maquinar provêm de um processo de fundição. Neste equipamento é feito o acabamento final da peça (facejamento, furação e roscagem). Para facilitar e diminuir o tempo de maquinação, cada mesa está equipada com fixadores hidráulicos e um bloco de aço com apoios para as peças (por norma duas ou quatro). O operador apenas necessita de colocar as peças nos apoios de forma correta e acionar um botão que ativa os fixadores hidráulicos, facilitando assim o processo de fixação e libertação. Esta forma de fixação apenas é viável porque o volume de peças iguais maquinadas é elevado e a precisão requerida é baixa quando comparada com maquinações CNC tradicionais. Anexo ao equipamento existe uma central hidráulica, cuja função é fornecer fluido hidráulico pressurizado a fixadores. A figura 2.3, é um esquema simplificado dos eixos da CNC *Chiron*. Como referido anteriormente, o fabricante do equipamento não é o mesmo que fabrica o sistema de controlo, daí o desvio da atenção do equipamento em si para o seu controlador.

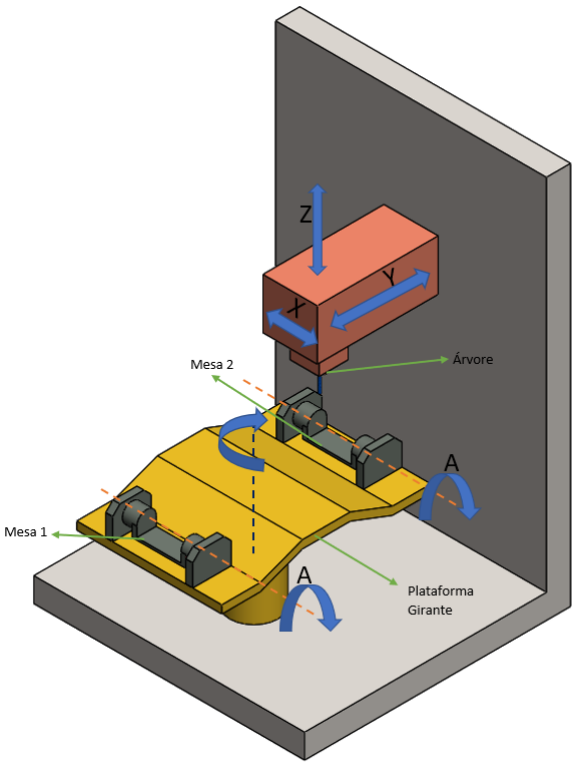


Figura 2.3: Esquema de eixos Chiron.

Processo de Maquinagem

Atualmente as ordens de produção são comunicadas ao colaborador através de uma aplicação em *Excel*, presente num computador na célula de produção e gerida pelo MES.

No início do turno, o operário identifica o número e o modelo de peças que deve maquinar. Neste momento é necessária a troca dos blocos das mesas para que coincidam com a peça a maquinar. Este é um processo ainda demorado, dado que o colaborador, com o auxílio de uma mini grua, manobra 4 blocos de elevada massa. Após inseridos os blocos, o operador seleciona o programa correspondente à peça e inicia a maquinação, sendo a sua função trocar as peças maquinadas por peças em bruto. Quando o número de peças de um determinado modelo atinge o valor requerido, o operador valida na aplicação o número de peças produzidas. Após a maquinação, as peças passam por uma estação de lavagem adjacente à CNC. A circulação das peças é feita com recurso a carrinhos com prateleiras compostas por cestas com fixadores de modo a garantir um transporte correto e livre de acidentes ou danos.

As peças maquinadas neste equipamento são feitas de uma liga de alumínio e atuam como *manifolds* de gás em esquentadores produzido na empresa, sendo responsáveis pela distribuição de gás no esquentador. Devido à sua funcionalidade, possuem furos com diâmetro muito reduzido (1 milímetro) e variados furos roscados onde serão posteriormente conectados assessórios de gás. A figura 2.4 mostra um exemplo de uma peça maquinada na *Chiron*.

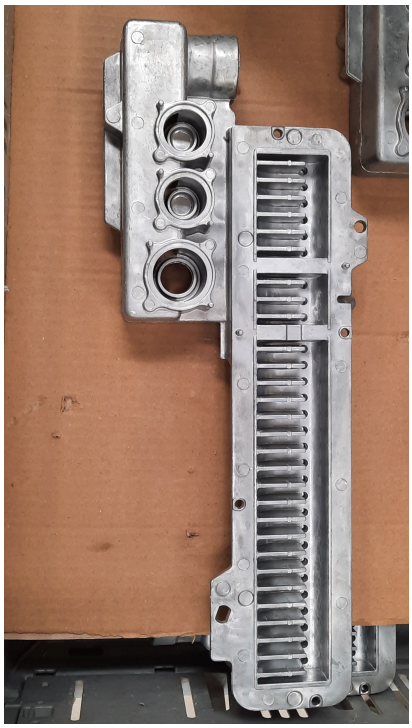


Figura 2.4: Peças maquinadas na CNC.

O controlo de qualidade é feito pelo mesmo colaborador que opera a máquina. Com um conjunto de equipamentos de medição, o operador compara as medida da peça maquinada com um intervalo de valores aceitável especificados num desenho técnico. Esta verificação é feita na primeira peça maquinada após a troca dos blocos fixadores e sempre que um número estipulado de peças é maquinado (entre 80 a 100). Se alguma medida estiver fora dos limites aceitáveis, a ferramenta responsável pela sua maquinação é substituída. No caso de uma ferramenta quebrar, todas as peças presentes na mesa de trabalho são enviadas para a sucata, o que, dependendo do componente, pode corresponder 2 ou 4 peças.

Capítulo 3

Revisão do Estado da Arte em Monitorização CNC

Este capítulo é dedicado à revisão do estado de arte, sendo que apresenta um resumo de toda a pesquisa efetuada em artigos, teses, produtos existentes e manuais que permitam a obtenção de um conhecimento mais aprofundado sobre os elementos constituintes do projeto e sobre possíveis soluções que dão que possam dar resposta aos problemas descritos anteriormente.

3.1 OPC

A sigla OPC, aquando da sua criação, agregava as letras iniciais de *OLE for process and control*, sendo por sua vez OLE a sigla para *Object Linking and Embedding*. OPC consiste num conjunto de normas e especificações que definem uma comunicação de dados segura, fiável e em tempo real entre equipamentos de diferentes fabricantes. Este conjunto de normas foi criado em 1996 por empresas no ramo de automação com o propósito de unificar os protocolos específicos de comunicação de Programmable Logic Controllers (Modbus, Profibus, etc.). Todas as regras, especificações, normas e princípios de funcionamentos são controlados pela OPC *Fundation* [14].

Um OPC atua como intermediário entre aplicações de alto nível HMI/SCADA (Supervisory Control And Data Acquisition) e os equipamentos a controlar, sendo a sua função converter pedidos de escrita ou leitura no formato requerido pelo equipamento de destino [15]. As especificações dos OPC Classic são baseadas nas tecnologias do Microsoft Windows COM (Component Object Model) e DCOM (Distributed Component Object Model), definindo separadamente o acesso a dados do processo (DA), alarmes (AE) e histórico de dados (HDA).

Anteriormente à criação do serviço OPC, sempre que uma aplicação necessitava de adquirir dados de um dispositivo de controlo era necessário desenvolver um *driver* especifico para a interface de comunicação com esse mesmo dispositivo, sendo que este processo se tornava exaustivo aquando da integração de múltiplos equipamentos com especificações de comunicação diferentes, ou dada a necessidade de múltiplas aplicações interagirem com o mesmo equipamento.

Com a introdução do OPC este processo tornou-se muito mais simples, pois após o desenvolvimento de um OPC *server* para um equipamento, qualquer aplicação que atue

como OPC *client* pode comunicar de forma indireta com o mesmo, facilitando assim a interação entre aplicações *Windows* e dispositivos de chão de fábrica.

Com o avançar dos anos a tecnologia OPC difundiu-se por toda indústria sendo que as suas normas são amplamente conhecidas. Por estes factos a OPC *foundation* decidiu mudar o significado da sigla OPC para *Open Plataform Comunications*.

A última geração de OPC designa-se por *OPC-UA* (*Unified Architecture*). Esta nova geração encontra-se amplamente difundida por fabricantes de equipamentos (enquanto *servers*) e fabricantes de sistemas Manufacturing Execution Systems e ERP (Enterprise Resource Planning) (enquando OPC *clients*).

OPC Server

Um OPC *Server* é uma aplicação *Windows* que tem como função a conversão do protocolo de comunicação de um equipamento, num protocolo de comunicação OPC, figura 3.1 [16].

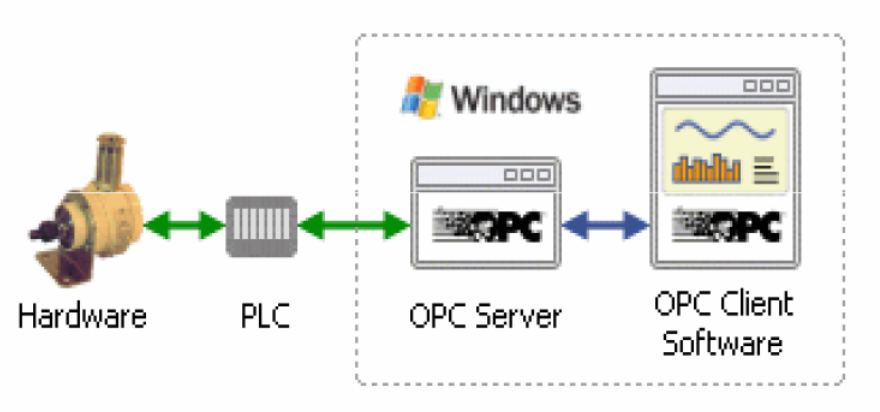


Figura 3.1: Arquitetura OPC [16].

Vantagens da utilização de um OPC:

- Baixo custo de integração devido à uniformidade do protocolo de comunicação.
- Eliminação da necessidade de utilização de *drivers* proprietários.
- Implementação de uma interface de comunicação única e consistente que permite um maior foco no desenvolvimento de novos recursos e funcionalidades em substituição do foco dado ao desenvolvimento de *drivers* de comunicação com dispositivos.
- Aplicações de controlo e monitorização podem atuar como OPC *Client* e *Server* permitindo uma maior cooperação entre aplicações.
- Integração com sistemas MES, ERP (Enterprise Resource Planning) e aplicações *Windows*.
- Facilidade de treino.

Note-se que o padrão OPC original foi desenvolvido para ser utilizado em ambiente *Windows*. O seu funcionamento em outros sistemas operativos requer a incorporação do serviço DCOM [17].

3.1.1 OPC Data Access

OPC *Data Access* são um conjunto de padrões para aplicações cliente-servidor que ditam regras de comunicação para a aquisição de dados em tempo real, atuando como intermediário entre sistemas de supervisão e os equipamentos de chão de fábrica. A aplicação servidor deverá conter os *drivers* de comunicação com os equipamentos.

Como apresentado na figura 3.2, um cliente OPC pode-se ligar a servidores OPC de diferentes fabricantes funcionando em simultâneo num computador ou em computadores diferentes. Com base no código fornecido pelos fabricantes de soluções OPC é possível determinar as especificações do mesmo.

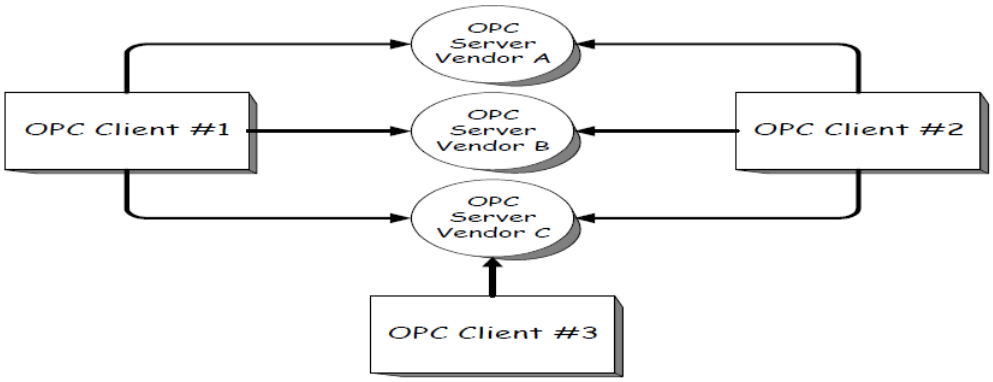


Figura 3.2: Interação cliente-servidor [18].

Um servidor OPC é formado por um conjunto de objetos:

- *Server*: Contêm a informação sobre o servidor e atua como contentor para os objetos do tipo *group*.
- *Group*: Contêm informação sobre si mesmo e atua com concatenador lógico e organizador de itens.
- *Item*: Representam uma ligação com uma fonte de dados dentro do servidor.

Este tipo de relação (figura 3.3) permite uma maior organização dos dados por parte dos clientes. Um *Group* pode ser do tipo *local* ou *public*. *Local* é quando pode ser subscrito por um só cliente e *public* quando está disponível para todos os clientes.

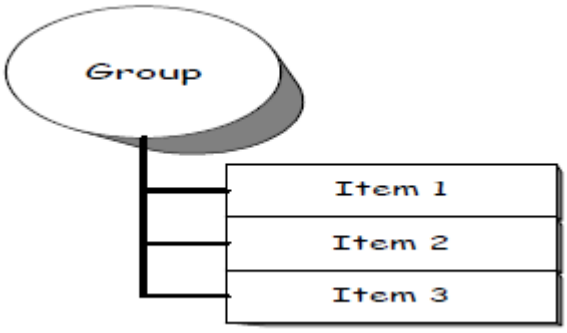


Figura 3.3: Relação Group/Item [18].

Um OPC *Item* só pode ser acedido através do objeto do tipo OPC *Group* que contêm o OPC *Item* desejado. Um OPC *Item* não é uma fonte de dados mas sim o endereço de uma. Associado a cada *item* surgem sempre 3 atributos [18]:

- *Quality*: 8 bits reservados para a codificação da qualidade dos dados enviados por uma fonte e podem ser de 4 tipos:
  - *Good*: Dado válido.
  - *Uncertain*: Impossibilidade em definir a qualidade do dado adquirido.
  - *N/A*: O atributo *quality* não é usado usado pelo OPC server.
  - *Bad*: Dado invalido.
- *Time Stamp*: Registo temporal, com base na estrutura *UTC (Universal Time Coordinated)*, do instante em que se deu a aquisição da informação. Quando esta informação não é disponibilizada pelos equipamentos de campo deverá ser gerada pelo servidor OPC.
- *Value*: Todos os dados do tipo *Variant data type (integer, bit, long, single, etc)* poderão ser empacotados e enviados.

Configuração dos dados OPC pelo Cliente
De modo a preservar os modelos dos *drivers* de comunicação específicos, os produtos de mercado limitam as configurações dos dados requeridos pelo cliente. Os clientes de dados OPC DA apenas podem definir as seguintes configurações [17]:

- **Criação de grupos e itens OPC**
- **Leitura Síncrona ou Assíncrona:** Para cada grupo OPC é possível exigir a confirmação da receção de uma leitura por parte do cliente antes de ser possível efetuar outra, i.e.,leitura síncrona. A leitura assíncrona não depende de nenhuma confirmação e normalmente garante um melhor desempenho.
- **Leitura de dados diretamente do dispositivo:** Esta opção permite a leitura dos dados diretamente do dispositivo de campo.

- **Estado Ativo ou Inativo:** Definição do estado de comunicação de um determinado item ou grupo, habilitando ou desabilitando o mesmo.
- **Leitura Cíclica ou por Mudança de Estado:** O cliente pode escolher ter os dados comunicados apenas quando se dá um mudança de estado ou receber os dados ciclicamente.
- **Banda morta:** O envio de dados pelo servidor ocorre se, e só se, o valor do dado adquirido estiver fora dos limites previamente estabelecidos para esse item.
- **Escrita Síncrona ou assíncrona**

Comandos de escrita e leitura funcionam separadamente. Após a receção de um comando de escrita, o servidor reencaminha-o instantaneamente para o correspondente dispositivo.

O padrão OPC recorre a uma comunicação por blocos de dados, garantindo uma otimização e redução do *overhead* da comunicação, uma vez que cada item é configurado como um só bloco de dados [17].

3.1.2 OPC Alarms and Events

Na década de 90, deu-se um grande aumento do nível de automação nas indústrias que se traduziu num fluxo crescente de informação a ser supervisionada pelos operadores fabris. A fim de alertar os operadores para ocorrências que requeiram imediata atenção, foram criados subsistemas que geriam alarmes e eventos. Deteção de eventos, erros e ativação de sistemas de emergências são exemplos das informações geridas por subsistemas desta natureza que, para além de alertarem operador, permitiam a recolha do histórico de funcionamento com o intuito de efetuar uma correlação posterior com outros dados.

As primeiras aplicações dedicadas a gestão de alarmes e eventos tinham por base softwares e sistemas de comunicação proprietários o que dificultava o acesso à informação por aplicações não consideradas no fase de desenvolvimento destas soluções. Como cada aplicação usava sistemas de difusão de informação diferentes tornava-se ainda mais difícil a integração de novas aplicações no fluxo de dados.

Como responsável pelos alarmes e eventos, a *OPC Foundation* criou em 1999 uma solução especifica para este valioso tipo de dados designada de *OPC AE*. A partir da implementação de um servidor *OPC AE*, desde que inclua os drivers de comunicação com os equipamentos de chão de fábrica, qualquer aplicação (*Windows*) que tenha a capacidade de atuar como cliente *OPC AE* pode subscrever aos alarmes eventos disponibilizados pelo servidor.

Um servidor *OPC AE* atua como camada intermédia. Por um lado possui a capacidade de adquirir os alarmes e eventos de um equipamento com uma comunicação específica, por outro lado oferece uma comunicação padrão a fim de disponibilizar os dados recolhidos com outras aplicações, facilitando o desenvolvimento de novas soluções de controlo e monitorização.

Quanto à sua complexidade, os servidores *OPC AE* podem desempenhar duas funções distintas:

- Adquirir alarmes e eventos de um só equipamento e transmiti-los a um ou mais clientes.

- Adquirir alarmes e eventos de vários equipamentos, através da capacidade de atuar como cliente de outros servidores ou recolhendo por si mesmo os dados, e transmiti-los a um ou mais clientes.

A subscrição de um servidor desta natureza tem por norma as seguintes finalidades:

- Informar o operador.
- Elaborar um histórico dos alarmes e eventos.
- Controlar subsistemas.

Normalmente é utilizado um *OPC AE* muito simples para detetar e comunicar os alarmes e eventos de um só equipamento, sendo esta a sua única função. Simultaneamente é implementado outro servidor mais complexo que atua como cliente dos servidores simples e executa todas as rotinas de controlo e gestão, a fim de disponibilizar informação mais organizada e interfaces mais complexas. A figura 3.4 esquematiza o fluxo de dados, desde equipamentos e módulos SPC (Statistical Process Control) até aos operadores, numa solução onde são utilizados vários servidores simples para recolha de dados e um servidor central dedicado à gestão dos mesmos.

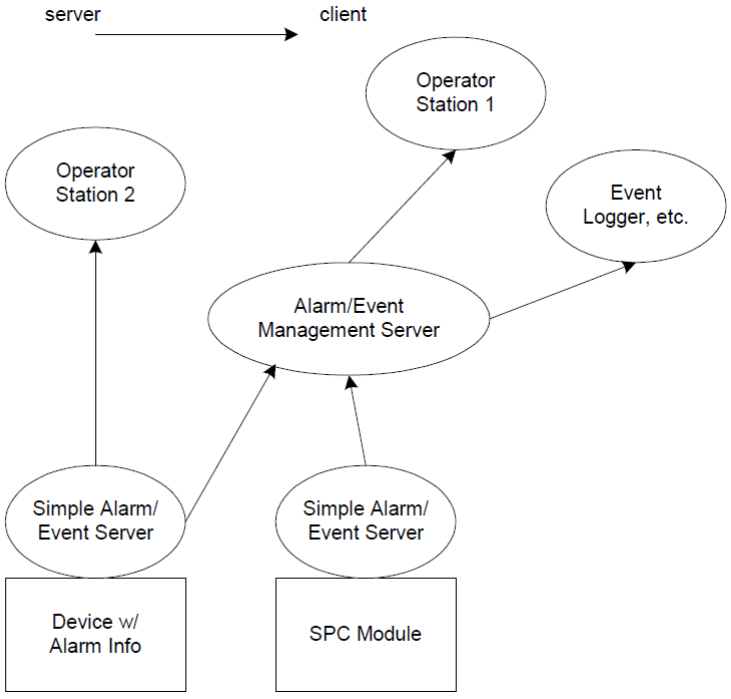


Figura 3.4: Interação entre vários servidores e clientes [19].

Neste panorama são designados de alarme todas as condições anormais de interesse especial. Já um evento poderá estar associado a uma condição ou a ações de operadores sensores, erros, etc. Um cliente OPC pode subscrever a notificação de vários eventos e alarmes. O servidor possibilita que um cliente:

- Identifique o tipo de eventos suportados pelo servidor.

- Subscreva eventos específicos ao servidor de modo a que seja notificado sempre que os mesmos ocorram.
- Atribuir *callbacks* no caso do encerramento de servidor *OPC AE*.

Uma condição é a designação do estado de um *OPC Event Server*, ou de um dos seus *OPC* itens caso este comunique com mais do que um equipamento. A estas condições dá-se o nome de *OPC Condition* e são sempre associadas a uma fonte (*OPC Source*). Uma condição pode possuir um ou vários estados. Uma condição multi-estados é sempre composta por um estado que engloba sub-estados (*OPCSubCondition*) mutuamente exclusivos. É exemplo de uma condição mono-estado uma falha *hardware*. Uma condição multi estados poderá ser um "*LevelAlarm*"composto por "*HighAlarm*", "*HighHighAlarm*"como exemplos de sub-condições. A arquitetura condição sub-condição tem por objetivo facilitar a identificação de eventos similares, dado que apenas uma sub-condição pode estar ativa. A figura 3.5 representa as interações entre fontes condições e sub-condições.

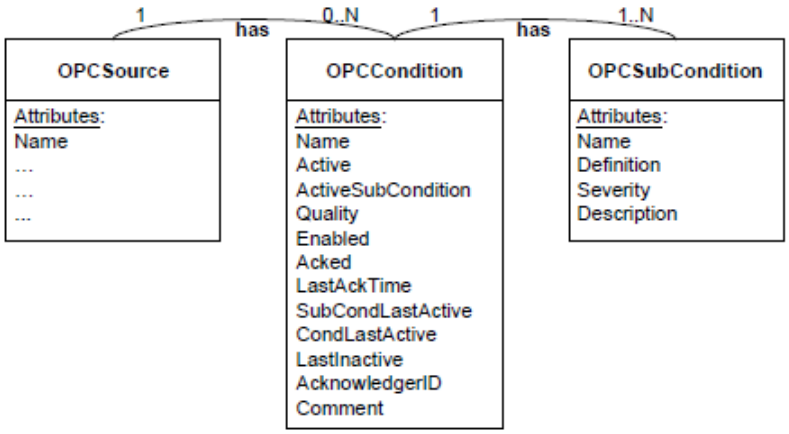


Figura 3.5: Interação entre fontes condições e subcondições [19].

OPCConditions

Cada condição possui os seguintes atributos:

- ***Name:*** Nome da condição, deve ser único no servidor.
- ***Active:*** O objeto representado está atualmente no estado designado pela condição.
- ***ActiveSubCondition:*** Sub-condição ativa.
- ***Enable:*** Condição atualmente supervisionada pelo servidor.
- ***Quality:*** Qualidade dos dados em que a condição se baseia.
- ***Acked:*** Condição confirmada.

- ***LastAckTime:*** Registo temporal da última *acknowledge*.
- ***SubCondLastActive:*** Registo temporal do último ocorrência da corrente sub-condição.
- ***CondLastActive:*** Registo temporal do último ocorrência da condição.
- ***LastInactive:*** Registo temporal da última transição para inativa da condição.
- ***AcknowledgerID:*** Identificação do último cliente a tomar conhecimento da condição.
- ***Comment:*** Comentário em formato *string* do último cliente a confirmar a condição.

OPCSubConditions

As sub-condições são compostas por:

- ***Name:*** Nome da sub-condição. Deve ser único entre as sub-condições associadas a uma condição. Caso se trate de uma condição de um só estado, o seu nome é igual ao da condição que lhe está associada.
- ***Defenition:*** Descrição que define o sub-estado representado pela condição.
- ***Severity:*** Gravidade da sub-condição.
- ***Description:*** Texto em formato *string* que é incluído na notificação do evento sempre que a sub-condição é ativa.

A severidade de uma sub-condição apresenta-se sob a forma de um número que representa a urgência da mesma. Este atributo pode variar entre 1 e 1000, sendo que 1000 é o caso mais grave e 1 o menos urgente. Por norma os servidores não possuem 1000 níveis diferentes de severidade, contudo a divisão em sub-escalas deverá ser feita em intervalos igualmente espaçados respeitando os valores da tabela 3.1 e desta forma possibilitando que clientes de diversos servidores fabricados por diferentes vendedores identifiquem o grau de severidade inequivocamente.

Tabela 3.1: Relação nível-valor de severidade.

Nível	Severidade
Elevado	667-1000
Médio	334-666
Baixo	1-333

Todas as notificações de eventos relacionados com mudanças de estado de condições que requerem confirmação por parte do cliente, incluem o nome da condição, o registo temporal da última ativação da condição ou da transição para a corrente sub-condição e uma *Cookie* única a cada notificação. Esta informação é utilizada pelo cliente na confirmação da condição. Se o cliente, devido a problemas de rede, confirmar uma condição que já tenha transitado de estado, o servidor ignora a confirmação apenas validando uma confirmação relativa ao estado atual da condição [19].

3.1.3 Eventos

Um evento é uma ocorrência importante para o servidor *OPC AE*, para o equipamento que lhe deu origem e para o cliente OPC. A ocorrência de um evento é transmitida por notificação a todos os clientes que subscreverem ao evento. O protocolo *OPC AE* divide os eventos em três categorias: simples, de rastreamento e de condição (alarmes).

Eventos Simples

Notificações de eventos simples são mensagens informativas que não necessitam de medidas corretivas. Notificações deste tipo contêm:

- ***Source:*** Identificação do objeto que gerou o evento e consequente notificação.
- ***Time:*** Registo temporal da ocorrência do evento.
- ***Type:*** Tipo do evento (simples, condição ou rastreamento).
- ***Event Category:*** Categoria a que o evento pertence, permitindo agrupar vários eventos com uma determinada categoria em comum.
- ***Severity:*** Severidade do evento.
- ***Message:*** Mensagem que descreve o evento.

Eventos de Rastreamento

Eventos de rastreamento indicam que uma ação especifica foi executada. São idênticos aos eventos simples na medida em que são informativos e não requerem nenhuma ação específica. Este tipo de notificações possui, para além dos atributos dos eventos simples, um *ActorID* que é responsável pela identificação do cliente responsável pela execução da ação que inicializou o evento.

Eventos de Condição

Eventos de condição (alarmes) ocorrem quando se dá a identificação de situações que requerem uma ação ou resposta. Estes eventos geram uma notificação ao cliente quando se dá uma transição do seu estado. Estes evento contêm os mesmos atributos que os eventos simples aos quais se anexam os seguintes:

- ***ConditionName:*** Nome da condiçã, e.g., Limite.
- ***SubConditionName:*** Nome da sub-condição ativa, e.g., HIHI/ LOLO.
- ***ChangeMask:*** Indica quais foram as propriedades da condição que se alteraram e levaram ao envio da notificação.
- ***NewState:*** Indica novos valores para as propriedades da condição (*Enable, ActiveeAcked*).
- ***Quality:*** Qualidade do dado transmitido pelo evento (*good, bad, uncertain,* ou *unknown*).

- ***AckRequired:*** Indica se é necessário uma confirmação.
- ***ActiveTime:*** Registo temporal do início da ativação do alarme.
- ***Cookie:*** Identificador único para cada notificação. É usado pelo cliente na confirmação do evento.
- ***ActorID:*** Identifica o *OPC client* responsável pela confirmação do evento.

A figura 3.6 apresenta de forma simplificada os estados e transições de estados associados a um evento de condição [20].

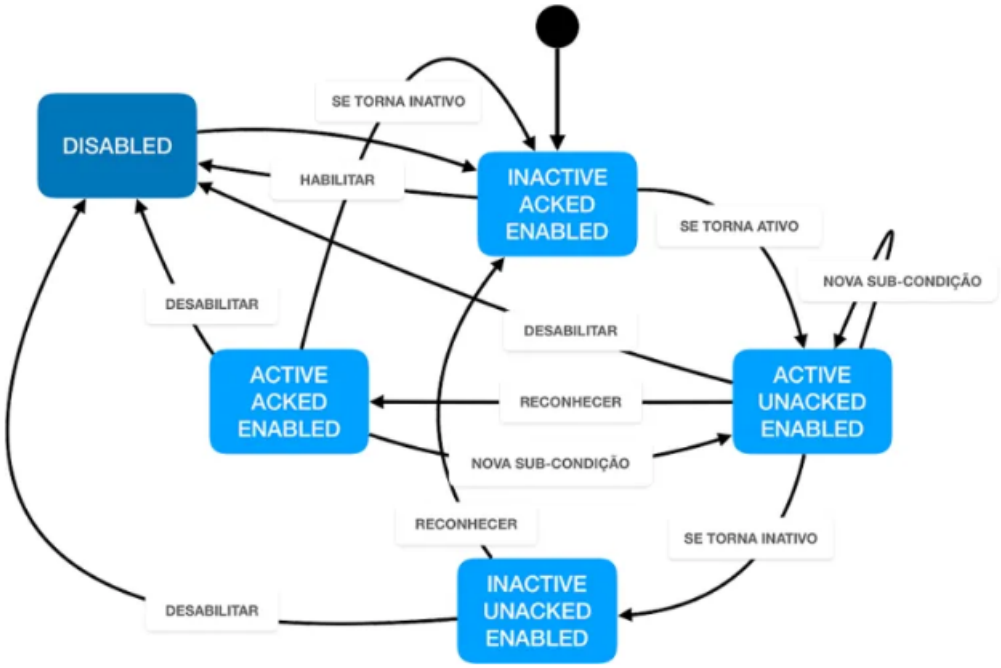


Figura 3.6: Transições de estado de uma condição [20].

3.2 CNC

Esta secção é dedicada a revisão de conceitos e ao funcionamento de sistemas CNC.

3.2.1 História

Com a revolução industrial surgiram as primeiras máquinas ferramentas. Estes equipamentos permitiam a produção de peças através da manipulação de volantes, alavancas e excêntricos. Apesar de se tratar de um trabalho repetitivo e tedioso, o uso destas máquinas permitiu a fabricação de peças de forma mais acelerada, económica e precisa, face aos métodos e técnicas anteriormente utilizados [21].

Desde cedo se verificou a necessidade de automatizar estes equipamentos. Uma das primeiras tentativas foi o uso de cames, que têm um funcionamento similar ao das caixas

de música. O maior entrave a esta tecnologia era o fabrico das cames em si, pois trata-se de um processo manual e demorado, sendo de elevada dificuldade o acerto do formato que permitisse uma produção precisa e sem falhas. Com a aplicação de hidráulicos nos sistemas de cames, foram desenvolvidos tornos copiadores. Estes equipamentos seguiam o relevo de uma peça modelo através do qual desbastavam um bloco de material até obter uma peça com a mesma geometria da peça original. Em 1950 a *General Motors* desenvolveu o que viria a ser a máquina ferramenta mais avançada da era pré-controlo numérico. Este equipamento registava os movimentos efetuadas pelo operador de modo a maquinar uma peça e depois repetia-os de forma automática e cíclica [22].

Os desenvolvimentos alcançados na área dos servomecanismos, na primeira metade do século 20, abriram caminho para o desenvolvimento dos NC (Numeric Control). Esta nova tecnologia e equipamentos, recorrendo a um ciclo fechado, possibilitaram um controlo preciso do movimento de motores. Este elevado grau de controlo é fundamental para o alcance de elevada precisão na automatização de máquinas ferramentas.

Na década de 40 a empresa *Parsons Corp* recebeu um contrato para o fabrico de hélices de helicópteros. A forma das hélices era definida através de um molde constituído por 17 pontos ligados por curvas francesas. Inicialmente estes moldes eram construidos em madeira, mas após uma série de problemas na construção das hélices, a empresa decidiu construir o molde em aço. Com recurso a uma calculadora de fita perfurada, foram gerados 200 pontos do complexo contorno, a partir dos quais foi maquinado manualmente o molde em aço. Este processo apenas definia as coordenadas dos pontos de formas complexas e mesmo utilizando um elevado número de pontos o relevo produzido era rugoso já que se tratava de um processo de maquinação manual.

Com o intuito de produzir uma máquina totalmente automática, *Parsons Corp* desenvolveu uma parceria com o *MIT*. Durante a segunda guerra mundial, o *MIT* tinha desenvolvido complexos sistemas de controlo de armas que foram implementados na máquina de *Parson*. A máquina resultante desta parceria era constituída por um computador que lia cartões perfurados e processava essas instruções, resultando no movimento automático dos eixos atuados por servo-motores [21]. Durante a década de 50, decorreram grandes avanços no NC, mas necessidade de criar fitas perfuradas ou magnéticas para cada peça, significava que a tecnologia não era viável na maioria dos casos. Com a introdução dos primeiros computadores, foi possível o desenvolvimento de programas que geravam automaticamente as fitas perfuradas com base numa lista de pontos e velocidades. Este avanço reduziu o tempo necessário ao desenvolvimento das fitas perfuradas mas continuava a existir uma separação entre o computador e o controlo numérico.

Em 1957, um grupo de entidades em parceria com o *MIT* associou um computador a um comando numérico, criando assim a primeira CNC. Para a programação deste equipamento foi desenvolvia a APT (Automatically Programmed Tool), esta linguagem padronizada baseava-se no léxico inglês facilitando a sua interpretação. A facilidade de programação e a diminuição dos preços dos computadores contribuíram para uma elevada difusão de *CNCs* na década de 60. O uso generalizado de equipamentos CNC revolucionou a indústria, na medida em que possibilitou a maquinagens de peças complexas de forma rápida e precisa. Outra vantagem foi o aumento da flexibilidade da produção de diferentes peças, já que se tornou muito mais fácil a troca de ferramentas e de programas peça.

3.2.2 Atualidade

Como se tratava de uma sistema proprietário, a linguagem APT foi entretanto substituída pelo *G-code* que é padrão na indústria. O desenvolvimento de um sistema CNC é uma tarefa dispendiosa e difícil, por isso, atualmente, os fabricantes de equipamentos CNCs adquirem os sistemas de controlo numérico de outras empresas. Por norma, os fabricantes e comerciantes dedicam-se ao desenvolvimento, construção e montagem do equipamento, desde a sua estrutura metálica até ao produto final, recorrendo a terceiros para a compra dos sistema de atuação e controlo. Esta divisão permite uma maior variedade e flexibilidade das características dos equipamentos comercializados. sendo que a atual configuração dos controlos numéricos permite uma elevada gama de implementações. Deste modo. possibilita aos clientes a escolha do controlo numérico para o qual estão mais familiarizados.

O desenvolvimento de sistemas de CAD (Computer Aided Design), CAM (Computer Aided Manufacturing) e DNC (Direct Numerical Control), tornam mais fácil e acessível todo o processo de maquinação, desde a elaboração do programa peça até a sua execução.

Atualmente com a digitalização da indústria, as máquinas *CNC* deixaram de ser um sistema fechado e passaram a permitir comunicações com outros equipamentos de chão de fábrica como robôs, AGVs (Automated Guided Vehicles) e também comunicações com sistemas de gestão e controlo fabril, o que proporcionam ferramentas para gerar quantificar OEE dos equipamentos a fim de melhorar o seu desempenho.

Nos dias que correm, praticamente todas as indústrias possuem equipamentos CNC, dado que as vantagens da sua utilização são indiscutíveis.

3.2.3 Código G

O código G é uma linguagem de programação utilizada no controlo de sistemas CNC. Trata-se de código compacto e de difícil interpretação pelo ser humano. A sua sintaxe é constituída por linhas onde são descritos códigos e posições que serão mais tarde interpretadas pela CNC. O código contém não só os movimentos a descrever, mas também a velocidade com que serão descritos, bem como outras funções complementares. Uma elaboração e implementação correta do código permite a transformação de um bloco de material na peça desejada. Em seguida é apresentado uma lista com os códigos mais comuns:

- **G:** Funções principais, sistemas de eixos, sistemas coordenadas, avanço linear ou circular, etc.
- **M:** Funções auxiliares, troca de ferramentas, controlo do liquido refrigerante, acionamento da árvore, etc.
- **F:** Função que define a velocidade de avanço.
- **S:** Função que define a velocidade de rotação da árvore.
- **H:** Função que define o comprimento da ferramenta
- **D:** Função que define o diâmetro da ferramenta.
- **T:** Função que define a ferramenta selecionada.

- **X:** Posição absoluta ou incremental de X.
- **Y:** Posição absoluta ou incremental de Y.
- **Z:** Posição absoluta ou incremental de Z.
- **N:** Número da linha.

Vantagens Em seguida são apresentadas algumas vantagens do uso de equipamentos CNC.

- **Precisão/Repetibilidade:** Reduzida diferença entre as dimensões da peça projetada e a peça produzida. Elevada consistência das dimensões das várias peças produzidas.
- **Fiabilidade:** Reduzido número paragens por falha do equipamento.
- **Diversidade/Flexibilidade:** Permite realizar peças diferentes desde que sejam da mesma família.
- **Produtividade:** Possibilita uma operação contínua, resultando numa grande quantidade de peças a um menor custo.

Desvantagens Desvantagens ao uso de CNC são enumeradas em seguida.

- Elevado investimento inicial.
- Necessidade de suporte técnico e manutenção qualificada.
- Necessidade de garantir um rendimento operacional mais elevado, sobe garantia que o equipamento não para.
- Necessidade de grande rigor no projeto do produto.

3.2.4 Arquitetura de um Sistema *CNC*

Um sistema CNC é constituído por vários componentes que permitem a leitura e a execução precisa do programa peça.

Entrada

A entrada não se trata tanto de um componente, mas sim da forma com que o programa peça é transmitido à máquina. Inicialmente esta operação era feita com recurso a um leitor fita, evoluindo para leitores de fita magnética. Com a intenção de melhorar a fiabilidade com que os programas eram transmitidos, foram desenvolvidos programas DNC (Direct Numerical Control). Estes programas permitiam a passagem, via *RS232*, de blocos do programa peça, de um computador para o equipamento, dado que a reduzida memória das CNC era incapaz de armazenar programas complexos. As máquinas atuais permitem a transmissão transmissão de programas via *USB* (*Universal Serial Bus*) ou rede *Ethernet*, ainda que interfaces *RS232* sejam muito utilizadas em máquinas mais antigas [23].

Numeric Control Kernel (NCK)

É a unidade central de qualquer CNC, tendo como funções a interpretação do código peça, gestão e controlo do sistema motriz, receção do sinal dos sistemas de medida e comunicação com o PLC.

PLC

O PLC é responsável pela atuação e leitura de todas as saídas e entradas de que são exemplo os botões de emergência, controlo do liquido de refrigeração e troca de ferramenta. Está em constante comunicação com o controlo numérico por computador.

Sistemas de Medida

Este sistema é constituído por transdutores que monitorizam continuamente a posição e velocidade da ferramenta. O controlo numérico recebe estes dados e compara-os com os especificados pelo código peça, efetuando correções se necessário.

Sistemas Motrizes

É constituído pelos servo-motores, fusos e amplificadores de sinal. Os sinais enviados pelo NCK para cada eixo são amplificados para poderem atuar os servo-motores, que por sua vez fazem rodar um fuso e consequentemente a peça ou a ferramenta mover-se-á.

Monitor (HMI)

O monitor é usado para a exibição de programas, permitindo o controlo do equipamento (transmissão e modificações de programas), comandos e visualização de parâmetros durante o funcionamento.

Funcionamento

Em seguida é descrito de forma resumida o funcionamento de um sistema *CNC*.

- Inicialmente é carregado o programa peça no NCK.
- Os dados do programa são processados. Interpolações ocorrem, se especificadas, seguido do envio de comandos para o sistema motriz.
- O sistema motriz movimenta-se de acordo com os comandos enviados pelo NCK. Simultaneamente, o PLC modifica o estado das saídas com base nas comunicações com o NCK.
- O sistema de medida adquire continuamente a posição e velocidade da ferramenta e envia-a para o NCK.
- O NCK compara os sinais do sistema de medição com os sinais enviados para os sistema de condução, e, se existirem erros, estes são corrigidos através do envio de um novo sinal para o sistema motriz.
- Um monitor é usado para a exibição dos comandos, programas e outras informações importantes.

3.3 Sistemas de Execução de Manufatura

A procura da perfeição é uma característica inerente à existência humana, concretizando-se num ciclo infinito de inovações, aperfeiçoamentos e melhorias sendo este parte integral da sociedade atual. Com a globalização da indústria, a concorrência empresarial aumentou exponencialmente. Para conseguirem competir neste novo mercado, as empresas necessitaram de implementar medidas de aumento de produtividade e diminuição de custos. Softwares de gestão industrial e planeamento foram desde muito cedo parte integral e fundamental no aumento do rendimento produtivo de recursos humanos e materiais.

Com a generalização do uso de computadores de grande porte na década de 40, surgiram os primeiros sistemas de informação implementados em atividades de produção. Estes sistemas estavam inicialmente ligados apenas ao departamento de finanças e contabilidade, avançando depois para produção, planeamento e controlo [24].

Criado na década de 60, o MRP (Material Requirement Planing) foi o primeiro sistema de gestão de recursos com uso generalizado na indústria. Este sistema permitia o cálculo da quantidade de materiais necessários e em que momento seriam precisos, levando a uma otimização da produção e a um maior controlo do inventário, culminando no aumento da produtividade e do rendimento. Com os excelentes resultados do MRP, surgiu a necessidade de expandir a sua capacidade para outros departamentos. Neste contexto surgiu na década de 80, o MRP II que incorpora, para além das funcionalidades do seu antecessor, recursos humanos, equipamentos, energia, recursos financeiros e *markting* [25].

Ambicionando uma integração de todas as componentes de um negócio no mesmo produto, surgiram na década de 90 os primeiros ERP (Enterprise Resource Planning). Neste contexto deu-se a necessidade de estruturar e definir os sistemas de informação. Tal documento foi elaborado em 2000 pela *Manufacturing Enterprise Solutions Association*, sob o nome de *ANSI/ISA-95*. Esta norma diferencia a hierarquia dos níveis associados a operação de produção, controlo de sistemas e outros sistemas de negócios, figura 3.7.

Figura 3.7: Hierarquia de uma organização segundo a norma ANSI/ISA-95 [26].

O nível 0 diz respeito ao processo produtivo. O nível 1 refere-se aos sensores atuadores, normalmente com uma latência de 1 segundo. Agrupados no nível 2 encontram-se os sistemas de automação e controlo do processo, normalmente PLCs, HMIs e SCADAs. O nível 3 (Manufacturing Operations Management) define as atividade que gerem a produção, incluindo também o registo de informações e a coordenação do processo. Por fim, do nível 4 fazem parte todas as atividades relacionadas com o negócio e gestão da empresa sendo neste nível que se enquadra o ERP. Os sistemas *MES* encontram-se maioritariamente no nível 3, executando uma ponte entre os níveis 0,1,2 e o nível 4 [26].

3.3.1 Funções do MES

A norma ISA–95.00.01 define um total de 12 função para o nível 3 no qual estão inseridos os sistemas *MES*. Tais funções são enumeradas em seguida.

- Distribuição de recursos e controlo.
- Distribuição da produção.
- Aquisição e armazenamento de dados.
- Gestão das operações de qualidade.
- Gestão do processo.
- Rastreamento de produção.
- Análise de desempenho.
- Agendamento e sequenciamento das operações.
- Controlo dos documentos.
- Gestão do trabalho.
- Gestão das operações de manutenção.
- Percurso, rastreamento e armazenamento dos materiais.

Um sistema MES necessita de um grau elevado de polivalência dada a necessidade de desempenhar múltiplas funções e de comunicar com todos os outros níveis. Para comunicar com vários níveis, equipamentos, sensores e atuadores é necessária uma flexibilidade nos seus módulos de comunicação, já que existe uma elevada diversidade da oferta para produtos destas categorias. No nível 3 estão presentes outros sistemas para além do MES *(computerized maintenance management system, warehouse management system, laboratory information management system)* aos quais este deverá ser capaz de transmitir informações e dados [27].

Enquanto o ERP tem uma visão macroscópica na organização, o MES atua mais ao nível microscópico entre ambos, existindo uma relação complementaridade. Ao efetuar a ponte entre os diversos níveis, o MES cobre a dificuldade do ERP em tratar o elevado volume de dados provenientes do chão de fábrica e a incapacidade de atuar na produção em tempo real. Ambos os sistemas possuem diversas funções, que permitem a sua utilização isolada em certas áreas e complementar em outras [28].

Em suma, um sistemas MES necessita de ser capaz de comunicar com todos os sistemas de uma organização, desde o chão de fábrica até ao ERP.

3.4 Soluções Existentes

Nesta secção são apresentadas soluções existentes no mercado para o projeto e pesquisas efetuadas sobre soluções implementadas.

3.4.1 *Development of Real Time Machine Tools Utilization*

O correto agendamento da manutenção de ferramentas utilizadas em equipamentos CNC deverá ser calculado com base nos dados de utilização de cada ferramenta. O cálculo deste valor é divido em *Power On*, fases de corte e fases em que a ferramenta está ativa, mas não se encontra em regime de trabalho.

A partir da conexão com sinais dos *encoders* associados a cada eixo, procedeu-se à recolha dos parâmetros de utilização de cada ferramenta. Com recurso a um sensor de corrente (*Adds on sensor*, figura 3.8) foi determinado o consumo do servo motor da árvore. O processamento dos dados adquiridos foi executado por um microcontrolador *WEMOS*, que transfere esses dados para um *broker MQTT* alojado num *Raspberry*. Este por sua vez insere os dados recolhidos numa base de dados. Este sistema permite uma visualização em tempo real dos parâmetros de utilização das ferramentas.

A extração do funcionamento dos eixos é processada pelo *WEMOS* que, com base nos sinais condicionados dos *encoders*, conta o número de pulsos e a direção do movimento, permitindo o cálculo da velocidade de rotação e de avanço da ferramenta. Simultaneamente os parâmetros de funcionamento da ferramenta são publicados via tópicos, num *broker MQTT*. Com recurso a um *dashboard* os dados monitorizados são expostos aos utilizadores.

A figura 3.8 representa em detalhe a arquitetura do sistema.

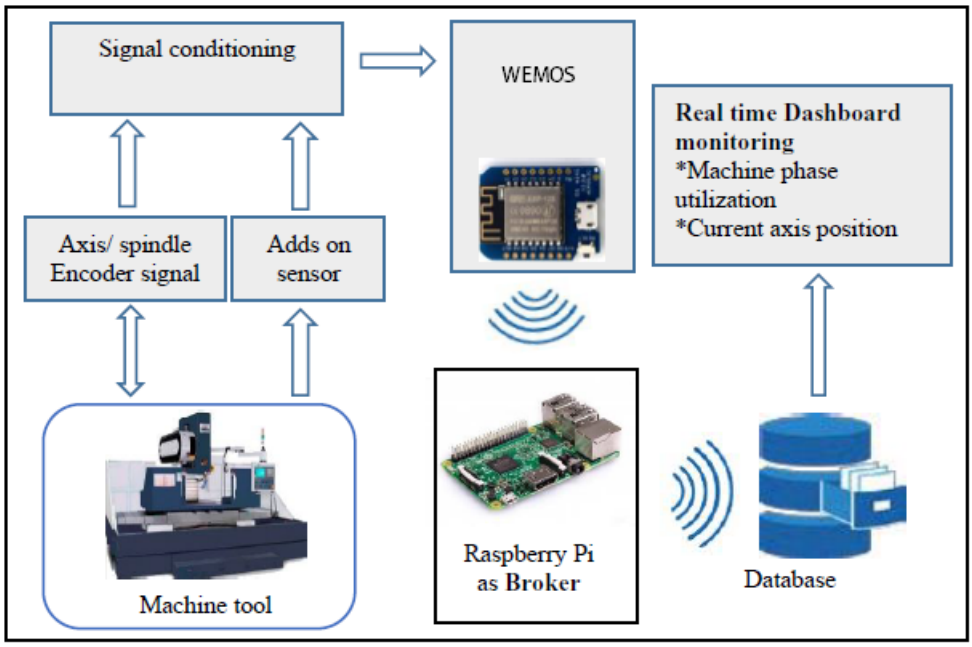


Figura 3.8: Arquitetura do sistema de aquisição de dados.

Dado que este sistema não interage com o unidade de processamento do comando numérico, mas sim com os seus *encoders* e sensores externos, a sua aplicação é muito abrangente, no que diz respeito à compatibilidade com diferentes comandos numérico, e não necessita de muitas adaptações aquando da sua implementação em comandos numéricos de diferentes marcas [29].

3.4.2 *Snap7 840d*

O PLC do controlo numérico *Siemens 840D*, em equipamentos fabricados após 2007, suporta vários padrões de comunicação, sendo o mais usado e versátil o protocolo de Ethernet TPCP/IP. Para comunicação com estes produtos, a *Siemens* disponibiliza ferramentas de controlo e monitorização, nomeadamente OPC UA. Estes produtos tem como desvantagem o limite de dados possíveis de monitorização e o seu elevado custo. Sendo que apenas possibilitam o acesso a variáveis do PLC. Um alternativa a este software é o *snap7*, uma biblioteca *opensource* de funções que permitem a escrita e leitura de DB (Data Block) presentes em *PLCs Siemens*. Esta biblioteca contém funções que possibilitam a conexão e desconexão com o PLC, escrita assim como leitura de bytes nos DB de memória do PLC.

O autor deste artigo, [30], desenvolveu uma aplicação que utiliza a biblioteca *snap7*, [31], de modo a estabelecer uma ligação com o PLC, monitorizar e modificar *bytes* de DB com a finalidade de obter o estado de funcionamento do centro de maquinagem.

Este sistema de controlo e monitorização remota para comandos numéricos *Siemens 840D sl* recorre à biblioteca *snap7* para extrair dados do PLC sendo comprovado por este artigo como fiável, e viável a sua utilização em sistemas mais complexos [30]. A desvantagem deste sistema em relação à utilização de OPC UA *clients* é o facto de apenas possuir a capacidade de extração de dados do PLC, enquanto o OPC UA permite para além dos dados do PLC, recolher variáveis do NCK (Numeric Control Kernel).

3.4.3 *Machine Metrics*

Machine Metrics é uma plataforma industrial IOT (Internet of Things) focada em monitorização de equipamentos CNC com o objetivo de melhorar a eficiência da produção do chão de fábrica. Esta solução, parte de um equipamento eletrotécnico que estabelece uma conexão com o centro de maquinagens, possibilita a aquisição de parâmetros como *Status*, *Modes*, carga, velocidade de corte e rotação e alarmes. Complementarmente são gerados um conjunto de *Web Services*, que permitem uma exibição em tempo real dos parâmetros adquiridos, monitorização do desempenho de cada centro de maquinagem e histórico de alarmes com vista à identificação de *bottlenecks* e diagnóstico de problemas. Como funcionalidade adicional é possível a integração vertical deste sistema com *ERPs*.

Com base nos dados obtidos, são elaborados *dashboards* com a finalidade de darem seguimento de trabalhos e peças, rasteamento de tempos de inatividade, informações dos operadores, desempenho de produção e é ainda fornecida a possibilidade de manutenção preventiva. Pela capacidade de comunicação com a maioria dos sistemas de controlo modernos, este produto é aplicável na maioria dos departamentos de produção.

Como exemplo de ganhos provenientes de mudanças efetuadas através da análise de informações disponibilizadas por este equipamento, é fornecido pela *Machine Metrics* o caso da *Fastenal*. Segundo esta empresa, com a introdução deste produto num espaço de três meses, recuperou trezentas e cinco horas de trabalho, as quais resultaram num

aumento do número de peças produzidas e numa maior percentagem de utilização dos centros de maquinagem. Tais aumentos encontram-se representados na figura 3.9 [32].

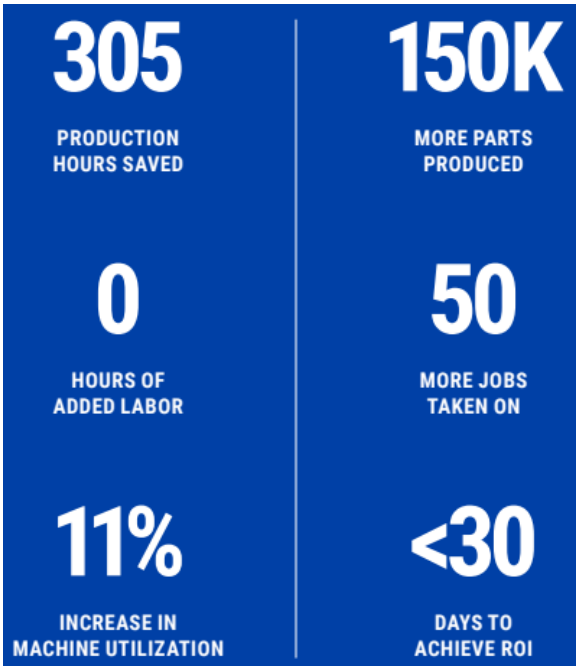


Figura 3.9: Ganhos com a utilização do *Machine Metrics* [32].

3.4.4 *Integration of Virtual and On-line Machining Process Control*

Segundo o estudo do autor, [33], nos anos recentes assistiu-se a um aumento do interesse em sistemas de maquinagem autónomos e auto-regulados. Para obter estes sistemas é necessário monitorizar o estado e condições de trabalho das ferramentas.

Tradicionalmente esta supervisão é implementada com recurso a sensores externos cujo a função é adquirir informação em tempo real sobre as condições de operação da ferramenta. A instalação destes sensores aumenta os custos de produção, e existe risco eminente de falha devido ao ambiente hostil em que são implementados. O maior problema destes sensores é a dificuldade de distinguirem entre fases de trabalho e fases de deslocamento em vazio, que culmina em falsos alarmes e má performance dos algoritmos de controlo baseados nos dados recolhidos.

Afim de resolver este problema o autor propõe a integração de um modelo virtual, representado na figura 3.10, do funcionamento do equipamento *CNC* nos algoritmos de controlo, permitindo separar os dados recolhidos em fases de trabalho dos restantes.

A aplicação desenvolvida, com base nos dados recolhidos e manipulados pelos algoritmos implementados, permite adequar os parâmetros de corte da ferramenta durante a execução do programa peça.

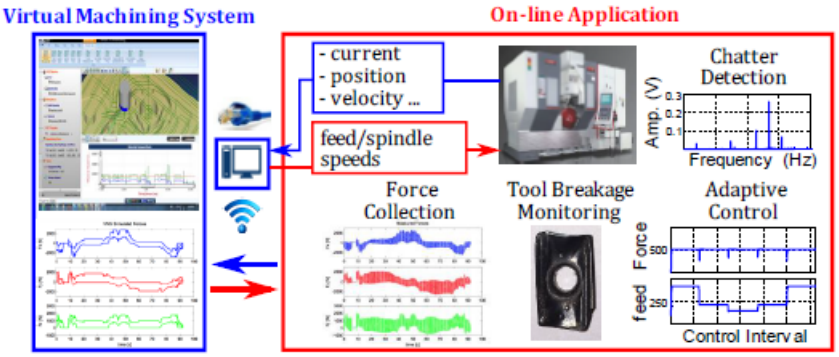


Figura 3.10: Sistema de controlo implementado [33].

Alterando na memória do *PLC* os valores de *override* da velocidade de rotação e de avanço é possível, em tempo real, atualizar as condições de trabalho sem que seja necessário alterar o programa peça.

A figura 3.11 representa os valores de força e avanço obtidos durante uma operação de fresagem executada sem controlo adaptativo (NoAC), com controlo adaptativo convencional (AC) e com controlo adaptativo assistido pelo modelo virtual (ACwithVF). Os resultados demonstram uma redução da força máxima de corte durante a operação, sendo que a maior vantagem será a adequação das condições de corte baseadas em dados de trabalho em tempo real, permitindo assim um aumento do tempo de vida das ferramentas e um melhor acabamento superficial. A integração do modelo virtual evidencia-se em zonas de iniciação de fases de trabalho, onde modelos convencionais têm dificuldade em identificar o momento exato onde se dá o início destas fases [33].

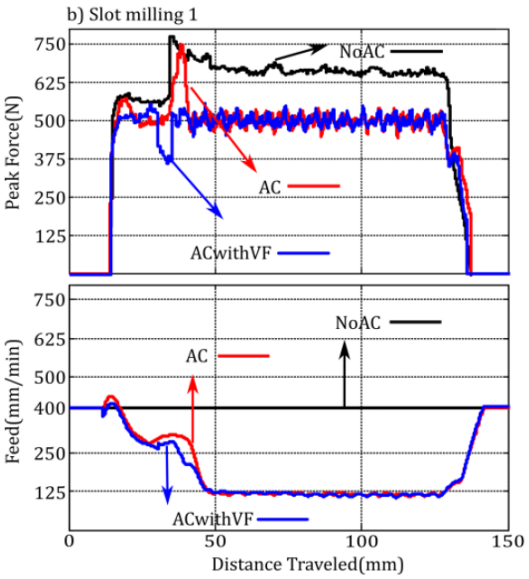


Figura 3.11: Resultados experimentais [33].

3.4.5 *uaGate 840D*

UaGate é um equipamento desenvolvido pela *Softing* que permite a leitura de dados presentes em *CNCs Sinumerik 840D sl*. Este dispositivo possui duas conexões *Ethernet*, uma para a ligação com o controlo numérico e outra para interface com *OPC UA* ou *MQTT* podendo ser usado com múltiplas aplicações de Indústria 4.0. A atuação como *firewall* é uma das funcionalidades deste equipamento, visto que possui duas interfaces independentes. A configuração dos dados recolhidos e da publicação dos mesmo torna-se simples pois recorre a um *web browser* gerido por um *web server* integrado no mesmo, não sendo requerido qualquer tipo de reprogramação do equipamento a monitorizar. Recorrendo a um servidor *OPC UA* interno este equipamento recolhe dados do *CNC* publicando-os simultaneamente num *broker MQTT* ou também num *OPC UA*, tornando-se de fácil integração com sistemas *SCADA*, ERP e MES. A sua robustez é comprovada pelos certificados de utilização em ambiente industrial que possui. Os dados recolhidos podem ser usados para a supervisão da, condições de trabalho, manutenção preditiva e rápida identificação de falhas no processo. A figura 3.12 representa o equipamento *uaGate*.

Figura 3.12: uaGate 840D [34].

3.4.6 *MtLink*

O *software Mtlink* é uma ferramenta disponibilizada pela empresa *Fanuc*, permitindo a construção de uma interface gráfica do *layout* dos equipamentos presentes num centro de produção através da qual é comunicada ao utilizador o estado de funcionamento dos equipamentos. Uma desvantagens deste sistema é a sua exclusividade de operação com equipamentos da marca *Fanuc*, sendo a facilidade de utilização e a compatibilidade com diferentes equipamentos o seu grande foco.

Este *software* permite a aquisição e consulta do histórico de desempenho e alarmes. Com base nos alarmes mais recorrentes e no desempenho individual de cada equipamento,

torna-se possível identificar e corrigir problemas que limitam a eficiência da produção. Um exemplo da interface do *MtLink* que demonstra a funcionalidade de aquisição dos resultados de produção, encontra-se representado na figura 3.13 [35].

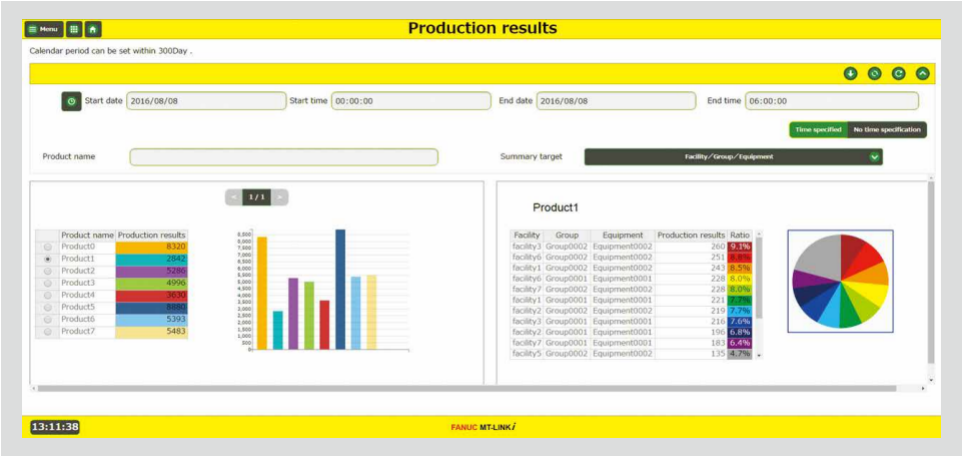


Figura 3.13: uaGate 840D [35].

3.4.7 *Research on Data Acquisition Numerical ControlMachine Tools*

Elevada precisão e eficiência são os símbolos de uma indústria moderna. Como os equipamentos *CNCs* são parte integrante das indústrias, a monitorização do seu estado, desempenho e precisão torna-se importante. Os autores deste artigo, [36], propõe a realização de testes circulares com o objetivo de testar a precisão destes equipamentos. O resultado deste teste permitirá realizar correções ao equipamento que aumentarão a sua precisão. Estes teste têm como base a comparação da trajetória circular ideal com a trajetória descrita.

Tendo como objeto de estudo um equipamento dotado do controlo numérico *Siemens Sinumerik 840D*, recorreu-se ao *software ePS network monitoring* presente no controlador. Uma interface MPI (Multi Point Interface Protocol) assente num DDE (Dynamic Data Exchange) *server* é utilizada na comunicação entre o *PLC* e a unidade de controlo numérico do equipamento avaliado. Com a alteração dos parâmetros do *ePs*, dá-se a recolha da posição da árvore e o simultâneo envio desses valores para um computador externo. A partir dos dados recolhidos é efetuada a análise do desvio de trajetória e apurada a precisão do equipamento, figura 3.14. A utilização deste sistema de recolha de dados permite recolher dados do *PLC*, *NCK*, *HMI* e dos *drivers* dos servo motores [36].

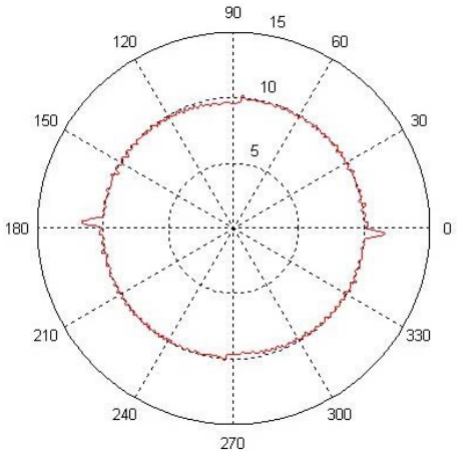


Figura 3.14: Trajetória discreta no teste circular [36].

3.4.8 Resumo das Soluções Existentes

As soluções previamente estudadas sofrem de uma (ou várias) das seguintes falhas:

- Incapacidade de envio de dados para o *Nexeed* MES: Uma falha comum em todas as soluções estudadas é a incapacidade, por defeito, de comunicar os dados recolhidos ao *Nexeed* MES, sendo necessária a implementação do processamento dos dados recolhidos, o que as torna impróprias para implantação na *Bosch Termotecnologia*.

- Reduzido número de dados recolhidos: A recolha do maior número de dados possível do equipamento é fundamental para o projeto. Só com este acesso é que é possível identificar quais os dados mais relevantes para o sistema MES. Soluções que passam pela utilização de sensores externos adicionados ao equipamento têm um número drasticamente reduzido de dados recolhidos. Por outro lado as soluções comerciais têm o seu *software* bloqueado para edição, apenas permitido a recolha dos dados com que vêm programadas.

- Incompatibilidade com o equipamento: Sendo o centro de maquinagem em estudo um equipamento antigo (*legacy*), o método de recolha de dados em algumas das soluções analisadas são incompatíveis com as tecnologias presentes no equipamento. As soluções [32], [34] e [35], utilizam tecnologias para a recolha de dados que simplesmente não são suportadas pelo equipamento em estudo.

- Baixa diversidade: As soluções analisadas focam-se num ramo específico do processo produtivo. Uma parte das soluções, aplica uma visão microscópica focando-se num detalhe do processo, utilização de ferramentas e controlo de posicionamento. A outra parte, aplica uma visão macroscópica focando-se apenas em contabilização de produção e tempos de funcionamento. Nenhuma possui uma abordagem flexível que permita uma conjunção das duas, fornecendo detalhes globais, mas com capacidade de se focar numa área do processo.

Apesar de já existirem várias soluções para a aquisição de dados e monitorização de equipamentos CNC, nada existe que possua a capacidade de adquirir, de forma fiável, dados do equipamento em estudo, conciliada com a capacidade de encaminhar esses mesmos dados para o sistema *Nexeed MES*. Será esta a premissa de todo o trabalho desenvolvido e apresentado nos próximos capítulos.

Capítulo 4

Solução Conceptual

De modo a estabelecer uma solução conceptual adequada ao problema exposto, é necessário considerar os requisitos impostos pela empresa e decorrentes da investigação, a fim de se chegar a uma solução que seja útil, flexível e adequada. Foi definido à priori que o sistema a desenvolver tem que ser capaz de recolher em tempo real o estado de funcionamento da máquina CNC, parâmetros de trabalho, programa executado, número de peças produzidas e ser capaz de comunicar com o sistema MES implementado.

Na figura 4.1 é representada a solução conceptual para os objetivos propostos. Nas secções seguintes será explicado em maior detalhe os diversos componentes da solução.

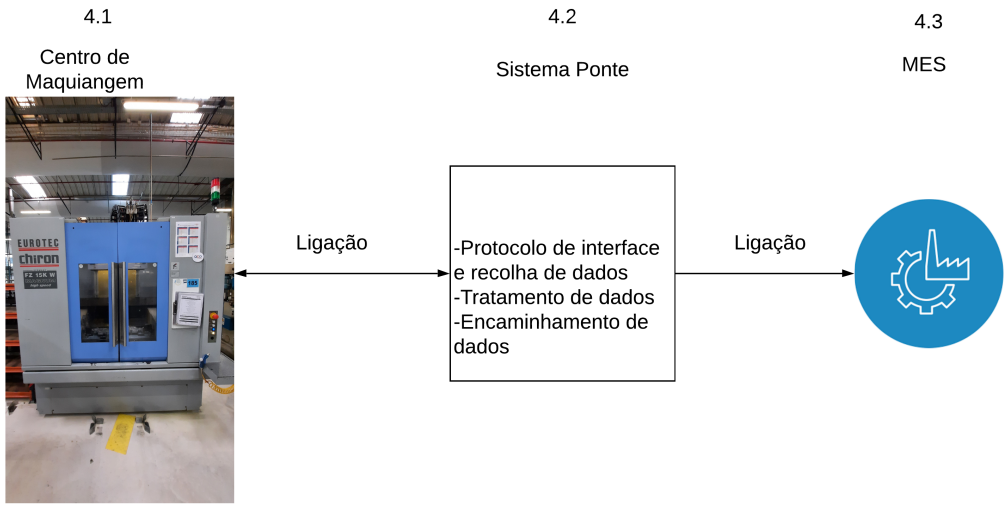


Figura 4.1: Solução conceptual.

4.1 CNC

O comando numérico computorizado a monitorizar é uma fresadora CNC *Chiron FZ15* implementada na linha de produção da *Bosch*, cuja função é a maquinação contínua de *manifolds* utilizadas no sistema de distribuição de gás de esquentadores pressurizados.

A cadência deste equipamento na produção de *manifolds* é de cerca de 240 peças por turno.

4.1.1 Siemens Sinumerik 840D

Sinumerik 840D é um sistema de controlo numérico computorizado fabricado e comercializado pela *Siemens*. O 840D que equipa a *Chiron* é uma versão de 2000, sendo quando comparada com o 840D sl, lançado em 2006, claramente rudimentar no que toca a digitalização e integração com a Indústria 4.0. As versões mais recentes deste equipamento possuem OPC UA e um sistema de comunicação *Siemens internet S7* que, em conjunto facilitam o acesso e disponibilização de dados do funcionamento.

A estrutura de *hardware* do *Sinumerik* é composta pelo HMI e o NCU (Numeric Control Unit) que é por sua vez constituído pelo PLC e pelo NCK *(Nmeric Control Kernel)*. O equipamento *SIMODRIVE 611D* encontra-se em comunicação constante com o NCU e é responsável pelo movimento e controlo dos servo-motores. O sistema é composto por 3 *hardwares* distintos, o *hardware* do HMI *Sinumerik PCU 50 5666MHz*, o *hardware* do NCU e o *hardwares* PLC. Cada um destes *hardware* executa aplicações e programas muito distintos. A memória do sistema encontra-se espalhada por EPROMs e RAMs (Random Access Memory) como é apresentado na figura 4.2.

A comunicação entre o PLC, NCK e o 611D recorre à comunicação DPR *(Dual-Port Ram)*. Este processo é possível devido a existência de vários canais de escrita e leitura disponibilizados pela RAM implementada. Mesmo possuindo um NIC *(Network Interface Card)* a comunicação entre o PCU (Computer Unit) e o NCU (Numeric Control Unit) recorre à interface MPI *(Multi-Point Interface)* que por sua vez é baseado no padrão *RS485* com instruções *Profibus-DP*.

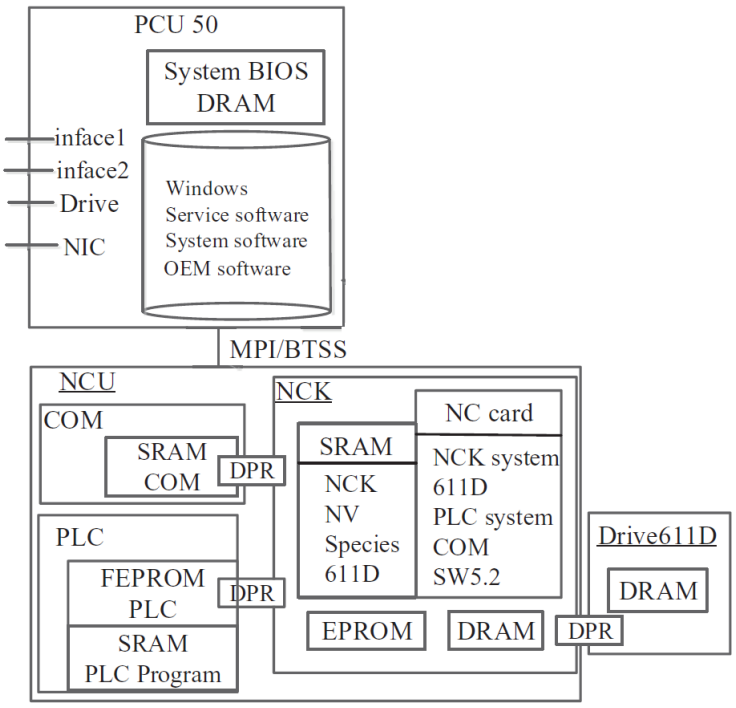


Figura 4.2: Estrutura do armazenamento de dados *Siemens 840 D* [37].

A comunicação de dados entre o HMI (PCU50) e o NCK utiliza a interface *NCDDE server*. Por sua vez, as aplicações presentes no HMI podem utilizar diferentes formas para comunicar dados com o servidor *NCDDE*. Recorrendo a uma comunicação direta com o servidor *NCDDE*, utilizando um *OPC Sinumerik server* como *middleware*, ou o serviço *DCTL-Control*.

O DCTL control é uma *ActiveX* usada na comunicação de dados entre servidor *NCDDE* e as aplicações do HMI. Usa recursos do *Windows* de forma a estabelecer uma estreita cooperação com o servidor e uma maior velocidade de transmissão em comparação com o lento serviço DDE [37]. A figura 4.3 descreve a comunicação entre aplicações presentes no HMI e o NCU.

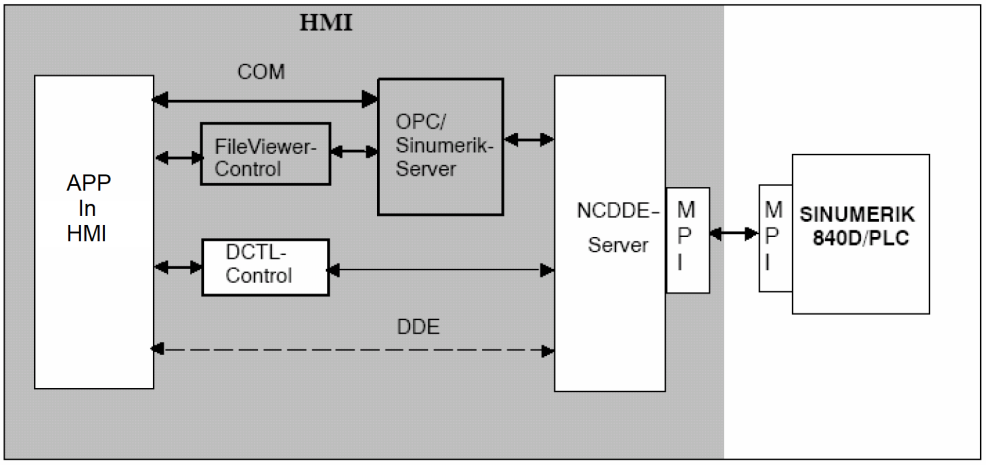


Figura 4.3: Comunicação entre aplicações HMI e o NCU.

4.1.2 Variáveis Controlo Numérico

Toda a informação necessária ao funcionamento do NCK é gerida por variáveis. Estas podem ser de vários setores, desde mensagens do PLC, parâmetros de corte, energia, alarmes, execução do programa peça etc. Algumas das variáveis do NCK podem ser acedidas pelo PLC e HMI. Este acesso pode permitir apenas a leitura ou também a alteração do valor da variável [38]. As variáveis estão organizadas em módulos de dados distribuídos por diferentes áreas.

As áreas utilizadas são:

- **NCK:** Variáveis que se aplicam a todo o NC.
- **Mode group (Bag):** Variáveis que surgem da ocorrência de diferentes estados durante o funcionamento.
- **Channel:** Todas as variáveis dos dados do sistema, áreas de proteção e estado global.
- **Tool:** Todas as variáveis que se aplicam às ferramentas na máquina.
- **Axis:** Contém os dados de configuração e os dados da máquina que se aplicam a cada eixo e à árvore.

- **Feed/Main drive (DriveVsa):** Contém parâmetros da máquina acerca do movimento da árvore.

Para além das áreas principais, existe uma subdivisão em módulos. Em cada área as variáveis são organizadas em estruturas ou tabelas. Para aceder corretamente a uma variável é necessário saber o seu endereço, que é constituído pela área, módulo, nome da variável e número da linha. Como exemplo, para aceder ao nome do programa da peça em execução é necessário o endereço */Channel/ProgramInfo/ProgName[u1]*. Neste caso a variável que contém o nome do programa encontra-se na área *Channel*, no módulo *ProgramInfo* e tem o nome de *ProgName*. Por fim, dentro de parênteses retos é necessário indicar qual o número da área, que neste caso a variável está no *Channel 1*, dai a indicação de *u1*.

Existem três tipos de variáveis: com apenas uma linha, com várias linhas e com várias colunas. No caso de variáveis de apenas uma linha podemos omitir o campo linha. Em variáveis multi-linha o campo linha necessita de ser especificado, juntamente com o número da área, como é exemplo */Channel/MachineAxis/actFeedRate[u1, 3]*. Quando a variável é composta por várias linhas e colunas, a sua indexação terá que identificar todo o caminho. Se quisermos saber qual a terceira função M ativa, temos que indicar */Channel/SelectedFunctions/Mval[u1,c1,3]*. Caso se pretenda recolher todas as funções M ativas, o caminho seria *[u1,c1,#5]*, sendo que a macro # indexa todos as linhas ou colunas até à especificada .

Todas as variáveis disponíveis ao utilizador encontram-se no manual *SINUMERIK 840D sl NC Variables and Interface Signals* [38]. Apesar deste manual se referir a um equipamento 840D mais recente contém não só as variáveis usadas na versão que equipa a *Chiron*, mas também outras que foram sendo adicionadas ao longo da evolução deste controlador numérico.

4.1.3 Programa PLC

O programa PLC do 840D está organizado por blocos: *organization, function, data* e *timer*. Este tipo de organização é utilizada pela *Siemens* na programação de todos os seus PLC. De fábrica estes blocos vêm programados e configurados, contudo possuem espaços vazios que poderão ser utilizados pelos fabricantes de equipamentos CNCs de modo a satisfazer as suas necessidades.

Organization Blocks (OBs)

Os blocos de organização são a interface entre o sistema operacional da unidade central de processamento do PLC e o programa do utilizador. São usados para executar secções específicas do programa como o arranque, erros e interrupções geradas pelo *hardware*. O sistema operacional do PLC executa o bloco de organização *OB1* ciclicamente. A partir dos blocos de organização podem ser chamados blocos de funções [39]. Outros blocos de organização estão programados para serem executados na ocorrência de um evento específico como no arranque do PLC.

A distribuição dos blocos de organização mais significativos é apresentada na tabela 4.1.

Tabela 4.1: Atribuição dos blocos de organização.

Número	**Função**
1	Execução cíclica
40	Processamento de alarmes
82	Diagnóstico de alarmes
86	Falhas nas cartas de entradas e saídas
100	*Start/Restart*

Data Blocks (DBs)

Os *Data Blocks* são estruturas de dados que podem ser usadas pelo programa para armazenar dados. Existem dois tipos de blocos de dados: DBs globais, onde todos os OB, FB (Function Block) e FC (Function Call) podem ler e guardar dados e blocos locais que são atribuídos um FB específico.

Toda informação trocada entre o PLC e o NCK é armazenada em *Data Blocks*. Estes DB contêm as informações relativas ao funcionamento do comando numérico computorizado. Um *DB*, pode conter até um máximo de 256 bytes. Cada *Data Block* pode ser construido por: *bits, bytes, words* ou *doubles*. A construção do programa PLC dita o formato com que uma variável é armazenada na memória. Este formato tem que ser usado aquando da sua indexação, por exemplo, caso se armazene uma *word* numa determinada posição do DB, os *bits* e os *bytes* que a formam deixam de ter significado quando lidos individualmente. A forma de indexação das variáveis dos *DBs* encontra-se representada na figura 4.4. Na tabela B.1 encontra-se representado uma visão geral dos DBs presentes neste equipamento

Bit								Byte	Word	Double	Data Block
7	6	5	4	3	2	1	0				
DB1.DBX 0.7	DB1.DBX 0.6	DB1.DBX 0.5	DB1.DBX 0.4	DB1.DBX 0.3	DB1.DBX 0.2	DB1.DBX 0.1	DB1.DBX 0.0	DB1.DBB 0	DB1.DBW 0	DB1.DBD 0	DB1
DB1.DBX 1.7	DB1.DBX 1.6	DB1.DBX 1.5	DB1.DBX 1.4	DB1.DBX 1.3	DB1.DBX 1.2	DB1.DBX 1.1	DB1.DBX 1.0	DB1.DBB 1			
DB1.DBX 2.7	DB1.DBX 2.6	DB1.DBX 2.5	DB1.DBX 2.4	DB1.DBX 2.3	DB1.DBX 2.2	DB1.DBX 2.1	DB1.DBX 2.0	DB1.DBB 2	DB1.DBW 2		
DB1.DBX 3.7	DB1.DBX 3.6	DB1.DBX 3.5	DB1.DBX 3.4	DB1.DBX 3.3	DB1.DBX 3.2	DB1.DBX 3.1	DB1.DBX 3.0	DB1.DBB 3			
DB1.DBX 4.7	DB1.DBX 4.6	DB1.DBX 4.5	DB1.DBX 4.4	DB1.DBX 4.3	DB1.DBX 4.2	DB1.DBX 4.1	DB1.DBX 4.0	DB1.DBB 4	DB1.DBW 4	DB1.DBD 4	
DB1.DBX 5.7	DB1.DBX 5.6	DB1.DBX 5.5	DB1.DBX 5.4	DB1.DBX 5.3	DB1.DBX 5.2	DB1.DBX 5.1	DB1.DBX 5.0	DB1.DBB 5			
DB1.DBX 6.7	DB1.DBX 6.6	DB1.DBX 6.5	DB1.DBX 5.4	DB1.DBX 5.3	DB1.DBX 5.2	DB1.DBX 5.1	DB1.DBX 5.0	DB1.DBB 6	DB1.DBW 6		
DB1.DBX 7.7	DB1.DBX 7.6	DB1.DBX 7.5	DB1.DBX 7.4	DB1.DBX 7.3	DB1.DBX 7.2	DB1.DBX 7.1	DB1.DBX 7.0	DB1.DBB 7			

Figura 4.4: Indexação de memória dos *DB*.

Function Blocks (FBs/FCs)

Os blocos FC e FB são utilizados para programar rotinas e instruções que, quando executadas, devolvem um ou mais valores de saída. A diferença entre um Function Block e um Function Call é a capacidade de reter os dados. Um FB pode conter memória interna e as suas variáveis são importadas de um data block. Em oposição, um FC contém memória interna temporária e as variáveis usadas possuem sempre o mesmo valor. Os blocos de funções FC e FB programados no *Sinumerik 840D* apresentam-se na tabela B.2 e B.3, respetivamente.

Timers

Como o próprio nome indica, são utilizados para cronometrar o tempo. Podem ser utilizados em vários contexto, como por exemplo atrasar a execução de uma tarefa. Os *Timers* encontram-se todos livres, tabela 4.2.

Tabela 4.2: Visão geral dos temporizadores.

Número	**Função**
1-512	Livre

4.2 Sistema Ponte

De modo a monitorizar o comando numérico é necessário desenvolver um sistema que atue como ponte entre a fresadora e o MES. A solução a desenvolver necessita de interagir com a fresadora de modo a permitir a aquisição de dados. Necessita também de processar os dados recolhidos, transformando-os em informações relevantes para a instalação fabril. Por fim, tem de ser capaz de comunicar com o sistema MES. O sistema deve ser reprogramável vem como robusto e fiável de maneira a tornar-se apropriado para utilização em ambiente industrial.

4.3 MES

Para a gestão do chão de fábrica está implementado um sistema MES que atua como intermediário entre os equipamentos e o ERP. Este sistema é utilizado como ferramenta de controlo da produção.

4.3.1 *Nexeed* MES

Nexeed MES, inicialmente designado de *OpCon MES*, é uma solução de gestão do chão de fábrica comercializada pelo Grupo *Bosch* e implementada na *Bosch Termotecnologia*. Com o objetivo de melhoria contínua dos processos, com recurso a histórico de dados, o *Nexeed Mes* garante a integração vertical, desde os sensores até à *cloud*. Como consequência, esta solução atua como ligação entre o sistema de controlo e os sistemas de *ERP*.

A solução ERP adotada é o *SAP*, que é a ferramenta ERP mais disseminada na indústria. Como solução de automação encontram-se utilizados equipamentos *Rexroth*. Ambas as soluções, *Nexeed* e *Rexroth*, são desenvolvidas pelo grupo *Bosch*, o que torna a implementação do *Nexeed MES* ainda mais simples e eficaz [6].

4.3.2 Módulos *Nexeed* MES

A figura 4.5 assemelha-se com a figura 3.7. No topo situa-se o *ERP*, neste caso o *SAP* e na base os equipamentos e processos produtivos e pelo meio encontra-se o *Nexeed MES*. A sua divisão por módulos permite forjar uma solução à medida de cada cliente. Em seguida serão analisados os diferentes módulos. Esta análise tem por base o documento *It Shopfloor Solutions* [40] e a informação disponibilizada *online* pela *Bosch* [6].

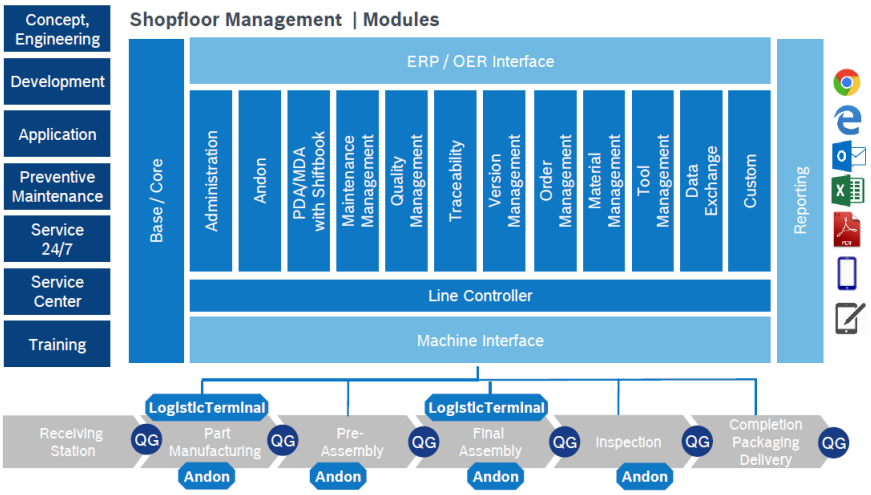


Figura 4.5: Módulos de gestão do chão de fábrica do *Nexeed* MES [6].

Base

Para a instalação do *Nexeed MES* são estritamente necessários módulos *Administration, Machine interface, Line controller* e *Reporting*. Recorrendo apenas a estes módulos é possível uma instalação e funcionamento básico do *Nexeed MES*.

Administration

O módulo *Administration* permite a criação de diferentes contas de utilizadores. Na especificação de cada conta está a área de informação que o utilizador poderá ter acesso e modificar. Esta característica é importante, pois permite um maior controlo e fiabilidade das informações e dados. O acesso a este módulo é feito através do portal *OIS.NET* acedido pelo *browser*.

Machine Interface

Machine Interface é sem duvida o módulo mais importante sem o qual seria inútil a função do *MES*. A sua função é estabelecer conexão de múltiplas fontes de informação (PLCs, computadores) com os outros módulos.

OIS.NET

O portal *OIS.NET* é o *frontend* do *Nexeed MES*. O seu acesso é feito com recurso a um *browser* onde é feita a *interface* com outros módulos (*Administration, PDA/MDA*).

ERP Connectivity

A comunicação com o *ERP* é uma funcionalidade extremamente importante que permite:

- Aquisição da estrutura de produtos.
- Informações do fornecedor.
- Planos de trabalho.

Reporting

Este módulo é baseado no portal *OIS.NET* e permite a geração de relatórios com possibilidade de exportação dos dados em vários formatos. Mais usuais, temos os relatórios de rastreabilidade, produção, qualidade e OEE.

Andon

O módulo *Andon* do *Nexeed* apresenta um elevado avanço na quantidade de informação em relação a sistemas *Andon* tradicionais de luzes. A informação disponibilizada por este módulo é comunicada, aos operadores, com recurso a *LCD*s posicionados numa zona de fácil visibilidade. As informações exibidas incluem o número de peças produzidas durante o corrente turno, o número de peças a produzir e a diferença entre ambos. A figura 4.6 representa um exemplo de um quadro *Andon*

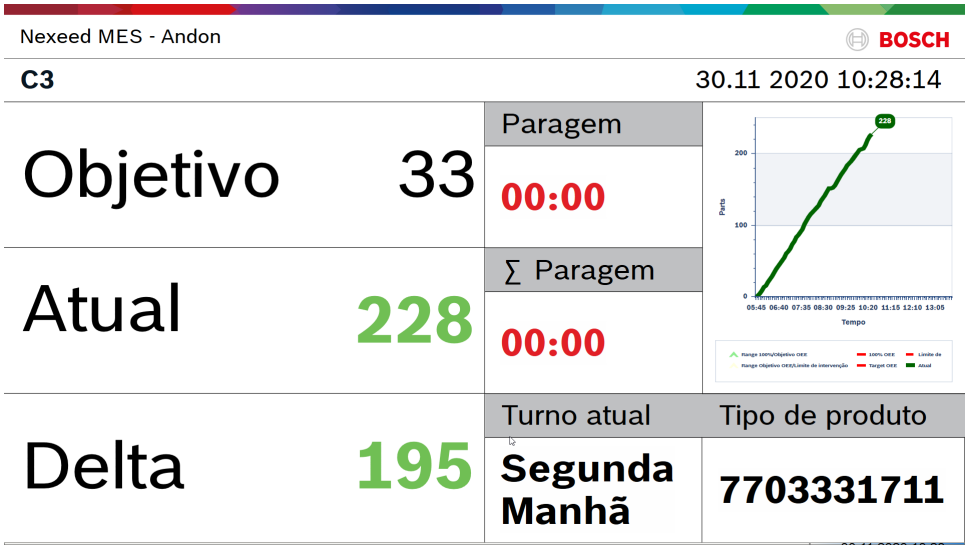


Figura 4.6: Andon [40].

PDA/MDA

O módulo PDA/MDA (Production Data Aquisition/Machine Data Aquisition) usa e processa grandes quantidades de dados da produção que são guardados na base de dados a fim de calcular *OEE*, *KPI*, número de partes produzidas e tempos de inatividade. O acesso a este módulo é feito com recurso ao portal OIS.NET.

Shiftbook

Shiftbook é um aplicação *Windows* cliente do serviço *PDA/MDA* onde são inseridos a duração dos turnos, períodos de inatividade, número de peças planeadas, peças produzidas e causas de inatividade de produção. O planeamento e organização de turnos e a documentação e gestão dos períodos de atividade são as principais funções deste módulo. A informação decorrente desta aplicações é posteriormente usado no *Andon*, cálculo de *OEEs* entre outros.

Material Control

Material Control torna possível a rastreabilidade do material. A ficha de produção de um produto passa também a conter os lotes de materiais que foram usados no seu fabrico. Se no futuro existir um problema originado em defeitos do material facilmente se identifica todos os produtos que utilizaram o lote defeituoso.

Tool Control

A função principal deste módulo é fornecer uma base de dados onde estão identificadas as ferramentas em uso, as suas características, número de ciclos efetuados e ciclos de vida. Com a sua utilização, torna-se possível a implementação de manutenção preventiva, tendo por conceito a troca de ferramentas baseada na sua utilização. Este tipo de manutenção permite um aumento do uso das ferramentas e uma diminuição de peças produzidas com defeitos, face a utilização de uma manutenção baseada em ciclos temporais.

Maintenance Management

Este sistema gere todo o processo de manutenção, pedidos de manutenção, gestão do armazém de peças, gestão dos recurso humanos da manutenção e armazenamento de histórico com informações relevantes sobre as manutenções efetuadas.

4.3.3 Arquitetura Nexeed MES

A implementação do *Nexeed MES* requer a utilização de 3 servidores independentes que comunicam entre si, como representado na figura 4.7.

O servidor *application* é responsável pela receção das comunicações com os equipamentos e pelos processos de computação dos módulos em uso. A comunicação entre este servidor e os equipamentos é feita principalmente recorrendo a telegramas XML (Extensible Markup Language).

Oracle server é um servidor de bases de dados onde são guardados os dados referentes à produção e está em constante comunicação com os outros dois servidores.

O portal *OIS.NET* é gerido pelo servidor *OpCon MES Web server* [41].

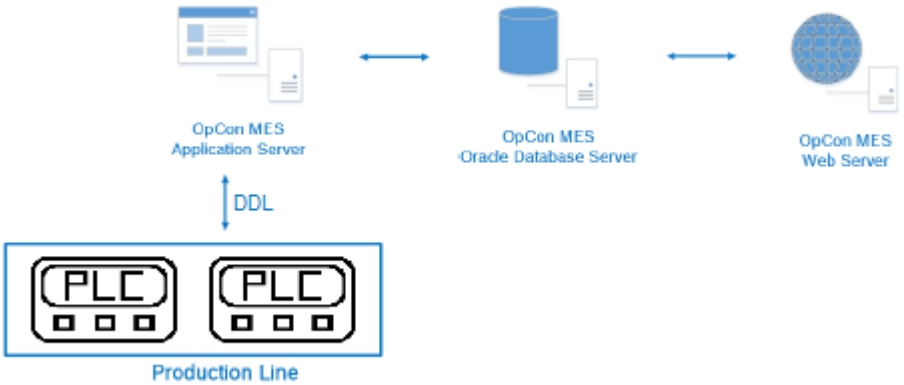


Figura 4.7: Arquitetura do sistema *Nexeed MES* [41].

Direct Data Link

Direct Data Link (DDL) é a designação da interface entre os equipamentos de chão de fábrica e o *Application Server* no sistema *Nexeed MES*. O DDL suporta comunicações via *OpCn XML protocol*, ficheiros .dat, *Opc Ua* e ainda um opção customizável. A diversidade de interfaces, representada na figura 4.8, permite a compatibilidade de comunicação com a maioria dos *PLCs* e componentes de controlo.

Considerando processos como atividades desempenhadas numa máquina, o serviço DDL recebe constantemente telegramas XML dos vários processos em curso e executa diferentes ações em função do telegrama recebido. Como exemplo, se o DDL receber um telegrama do tipo *partProcessed* os dados contidos no telegrama são armazenados nas bases de dados do *PDA/MDA* e da Qualidade [41].

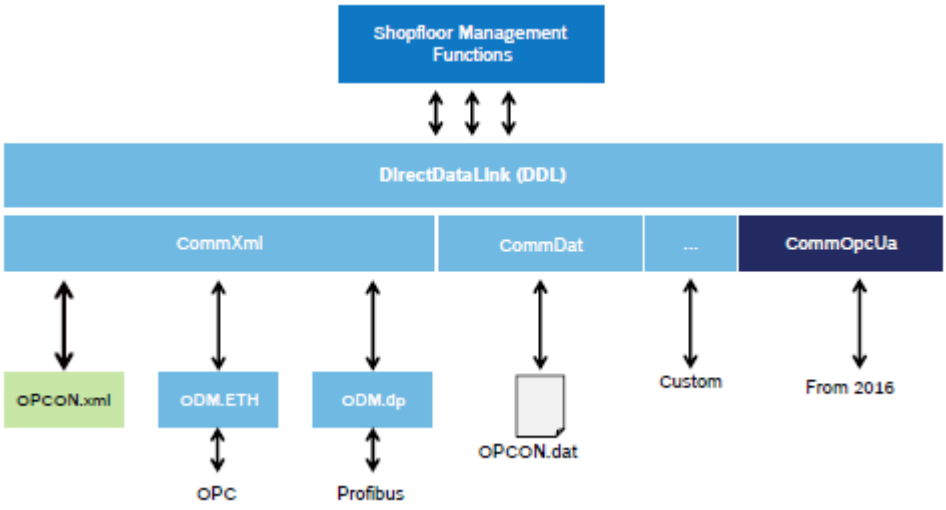


Figura 4.8: Interface com os equipamentos [40].

4.3.4 OpCon XML protocol

Como referido anteriormente, a troca de dados entre o ecossistema *Nexeed MES* dá-se principalmente com recurso a telegramas que se regem pelo protocolo *OpCon XML Protocol* [42]. Em seguida é explicada a estrutura dos telegramas com base no descrito pelo documento OpCon Xml Specification V2.2, elaborado pela *BOSCH Connected Industry* [42].

XML Protocol

Extensible Markup Language é uma linguagem que define um conjunto de regras para a codificação de dados de forma a possibilitar a sua transmissão e armazenamento. Tratando-se de uma *markup language* um ficheiro escrito sob este formato utiliza *tags* para descrever o seu conteúdo, figura 4.9. O conteúdo de um documento *XML* é composto por elementos, que, por sua vez, são constituídos por uma *tag* de início, *tag* de conteúdo, e uma *tag* de fim. Os atributos são normalmente utilizados para fornecer informações adicionais que não fazem parte dos dados. Um elemento pode ainda conter

vários elementos.

Hoje em dia esta linguagem é comummente utilizado na indústria como forma de transmissão de dados e estruturas de dados devido à sua elevada portabilidade e por não necessitar de um *software* específico para a sua escrita ou leitura [43].

Na figura 4.9 é apresentado um exemplo de um ficheiro escrito sob o formato *XML*.

```
<?xml version="1.0" encoding="utf-8" standalone="yes"?>
<note>
  <to AttributeGender = "male">Helton</to>
  <from AttributeGender = "female">Jani</from>
  <heading>Reminder</heading>
  <body AttributeType = "letter">
  Don't forget me this weekend!
  </body>
</note>
```

Figura 4.9: Exemplo de um documento XML.

Transmissão

Os telegramas são transportados usando a família de protocolos TCP/IP (Transmission Control Protocol - Internet Protocol). Adicionalmente, é anexado à mensagem um cabeçalho de 4 *bytes* que contêm o tamanho total da mensagem a ser transmitida (corpo e cabeçalho). Este método de transmissão não restringe o tamanho da mensagem. No caso do tamanho da mensagem ser superior ao tamanho máximo de dados encapsulados por esta família de protocolos, esta será dividida em várias mensagens na fonte e depois reagrupada no destino.

Estrutura

OpCon XML é um padrão que define normas para a elaboração de telegramas em formato *XML* de forma a que sejam corretamente transmitidos e interpretados pelo serviço *DLL*. O protocolo foi desenvolvido com o intuito de abranger o maior número de equipamentos de produção possível. Equipamentos simples como PLCs poderão apenas suportar as especificações básicas, sendo que outros sistemas mais poderosos (computadores industriais) podem tirar partido de todas as funcionalidades.

A primeira linha de um documento XML é uma declaração que identifica o documento como sendo do formato *XML*, a versão do protocolo XML usada (*Version*), o tipo de codificação utilizada (*Encoding*) e ainda a dependência do documento de fontes externas (*Standalone*). Não é obrigatório a implementação da declaração mas se usada necessita obrigatoriamente de conter o atributo da versão, enquanto o *Encoding* e o *Standalone* são opcionais [44].

No que diz respeito ao *OpCon XML protocol*, o corpo do documento é constituído pelo elemento `<root>` que contêm 3 elementos filhos, como o que se encontra apresentado na figura 4.10. Os elementos dentro do `<event>` e do `<body>` podem variar, o `<header>` possui uma estrutura fixa. É obrigatória a utilização do `<header>` e do `<event>` sendo que o `<body>` é opcional e pode ser configurado de acordo com as necessidades do

cliente. Esta estrutura é independente do tipo de informação a transmitir e, é utilizada nas mensagens enviadas, recebidas e interpretadas pelo MES.

```
<?xml version="1.0" encoding="UTF-8"?>
<root>
  <header/> <!-- mandatory -->
  <event/>  <!-- mandatory, event-specific structure -->
  <body/>   <!-- mandatory, application-specific structure -->
</root>
```

Figura 4.10: Estrutura base de um telegrama OpCon XML [42].

Header

O elemento `<header>` é constituído por atributos que contêm informações sobre a mensagem e permitem uma identificação exclusiva da mesma. Estes atributos estão listados na tabela 4.3.

Tabela 4.3: Atributos do elemento *header*.

Atributo	Especificação	Formato	Descrição
eventId	Obrigatório	UDINT	Identificador único de um telegrama de um evento. Telegramas de pedidos e respostas têm o mesmo *eventId*.
eventName	Obrigatório	STRING	Nome do evento, determina a interpretação da mensagem e o processamento efetuado pelo *MES*.
version	Obrigatório	STRING	Versão do protocolo *OpCon XML* em uso.
eventSwitch	Opcional	DINT	Usado para controlar a seleção do processamento correto quando vários processos são possíveis para o evento.
timeStamp	Opcional	STRING	Data e hora do envio do telegrama em formato *W3C(World Wide Web Consortium)*.
user	Opcional	STRING	Código do utilizador.
pwd	Opcional	STRING	Palavra passe codificada do utilizador.
contentType	Opcional	DINT	Caracterização do tipo de formato conteúdo(items, arrays, structs, structArrays e trace).

Para além dos atributos, o elemento `<header>` contém um elemento filho obrigatório designado por `<location>`. A função deste elemento é caracterizar a localização do evento que culminou no envio do telegrama. Os atributos da `<location>` são apresentados na tabela 4.4.

Tabela 4.4: Atributos do sub-elemento *location*.

Atributo	Especificação	Formato	Descrição
lineNo	Obrigatório	UINT	Número da linha
statNo	Obrigatório	UINT	Número da estação
statIdx	Obrigatório	UINT	Índice da estação
application	Obrigatório	STRING	Nome do dispositivo que enviou o telegrama
fuNo	Opcional	UINT	Índice de unidade funcional
workPos	Opcional	UINT	Posição do trabalho
toolPos	Opcional	UINT	Posição da ferramenta
processNo	Opcional	UINT	Número do processo
processName	Opcional	STRING	Nome do processo

Na figura 4.11 está apresentado um exemplo para o elemento `<header>` de um evento do tipo *partProcessed* decorrente de um ensaio de fugas e proveniente de um PLC.

```
<?xml version="1.0" encoding="utf-8"?>
<root>
  <header eventId="2" eventName="partProcessed" version="1.0" eventSwitch="-1" contentType="3">
    <location lineNo="1" statNo="40" statIdx="1" fuNo="1" workPos="1" toolPos="1" application="PLC"
    processName="C7 Water Leakage Test" processNo="1040" />
  </header>
```

Figura 4.11: Exemplo de um *header*.

Event

O elemento `<event>` contém toda a informação específica ao evento que originou o envio do telegrama. É formado por um ou mais sub-elementos que por sua vez contêm atributos obrigatórios e opcionais específicos ao sub-elemento. Os sub-elementos passíveis de utilização encontram-se expostos na tabela A.1.

Body

Por fim o elemento `<body>` contêm dados específicos a um evento. Ao contrário dos outros dois elementos anteriores, os sub-elementos deste elemento são de livre configuração de maneira a adaptarem-se às necessidades específicas a cada evento. Se não existir a necessidade de acrescentar mais informação no telegrama este elemento pode estar vazio.

Items

O elemento `<items>`, figura 4.12, apenas pode ser utilizado como sub-elemento direto do `<body>` e é composto por um ou vários `<item>`. Por sua vez cada `<item>` contém um parâmetro (variável de arranque, medida, etc) constituído por 3 atributos obrigatórios:

- *name (STRING):* Nome do parâmetro
- *value (Variável):* Valor do parâmetro

- *dataType (DINT):* Tipo de dado em que o *value* foi escrito (DINT, INT, STRING, real, etc)

```
<items>
    <item name="identifier" value="5572275000018773160061" dataType="8" />
    <item name="typeNo" value="773160061" dataType="8" />
    <item name="printLabelFilename" value="CartonLabel" dataType="8" />
    <item name="printerName" value="printer1" dataType="8" />
</items>
```

Figura 4.12: Exemplo de um sub-elemento *item*.

Arrays

O elemento `<arrays>` apenas pode ser utilizado uma vez no `<body>`. Este elemento é composto por sub-elementos `<array>` e é utilizado para transmitir arranjos no telegrama. Os `<array>` contêm os atributos *name* e *dataType* que classificam o nome do arranjo e o formato dos dados, respetivamente. Os elementos dos `<array>` são do formato `<item>` e contêm os valores do arranjo. A figura 4.13 é exemplo da implementação de um `<arrays>`.

```
<body>
    <arrays>
        <array name = "amostra" dataType="5">
            <item value = "0.0"/>
            <item value = "1.1"/>
            <item value = "2.2"/>
        </array>
        <array name = "teste" dataType="3">
            <item value = "0"/>
            <item value = "11"/>
            <item value = "22"/>
        </array>
    </arrays>
</body>
```

Figura 4.13: Exemplo de um sub-elemento *arrays*.

StructArrays

O elemento `<structArrays>` é o mais versátil de todos, pois permite uma configuração livre dos dados a transmitir. Os sub-elementos do `<structArray>` são do tipo `<array>`. Cada `<array>` precisa obrigatoriamente de conter o atributo *name* (nome do arranjo) e dois sub-elementos, o `<strutcDef>` e o `<values>`. O `<strutcDef>` é formado por elementos do género `<item>` que contêm a estrutura do tipo de dados que constitui o arranjo. O elemento `<values>` contém os dados do arranjo.

Como mostra a figura 4.14, no elemento `<structDef>` são definidos os nomes e o tipos de dados usados nos atributos que formam os sub-elementos do elemento `<values>`. O elemento `<values>` é composto por `<item>` que descrevem os elementos individuais do arranjo.

```
<body>
    <structArrays>
        <array name = "TestInfo">
            <structDef>
                <item name="name" dataType="8" />
                <item name="value" dataType="8" />
                <item name="unit" dataType="8" />
            </structDef>
            <values>
                <item name="Calibrated Leak" value="Skip" unit="Pa" />
                <item name="Leak" value="pass" unit="Pa" />
                <item name="Pressure Check" value="pass" unit="Pa" />
            </values>
        </array>
        <array name = "TestResults">
            <structDef>
                <item name="name" dataType="8" />
                <item name="value" dataType="4" />
                <item name="unit" dataType="8" />
            </structDef>
            <values>
                <item name="Time" value="150.3" unit="seconds" />
                <item name="Angle" value="540.32" unit="degrees" />
                <item name="Torque" value="2.755" unit="Nm" />
            </values>
        </array>
    </structArrays>
</body>
```

Figura 4.14: Exemplo de um sub-elemento *structArrays*.

resHead

O elemento `<resHead>` tem uma estrutura definida à semelhança do `<header>` e apenas poderá aparecer uma vez no `<body>`. As informações contidas neste elemento visam transmitir uma breve visão geral do resultado do processo e dados básicos sobre a peça. Dependendo do atributo *contentType* do elemento `<header>`, o `<resHead>` poderá ser inserido como sub elemento de uma `<struct>`. Nada é definido acerca da estrutura de uma `<struct>` sendo livre a sua composição, porém deve-se garantir que quem envia e recebe a mensagem conhece e è capaz de interpretar os seus elemento. Na tabela 4.5 estão descritos todos os atributos que podem ser utilizados no `<resHead>`.

Sequência de comunicação

Sempre que um evento ocorre numa estação de produção, este é transmitido ao MES num telegrama *XML*. O MES não confirma a receção, sendo esta assegurada pelo protocolo de comunicação. Em seguida o MES começa a processar o evento de acordo com as configurações de processamento para eventos daquele tipo. No final do processamento o MES, se assim configurado, responde com um envio de um telegrama que é composto pelo mesmo `<header>` do telegrama recebido e pelo resultado do processamento. Este resultado é transmitido recorrendo aos sub-elementos `<result>` e `<trace>` do elemento `<event>`. Se for necessário transmitir mais informações à estação, estas serão anexadas no `<body>`.

Tabela 4.5: Atributos do elemento *resHead*.

Atributo	Especificação	Formato	Descrição.
Result	Obrigatório	short	Resultado do processos, 1 se OK, 2 se NO
NioBits	Obrigatório	DINT	Número identificativo do erro ou erros *(bit-coded)*, se Result=1, Niobits=0
TypeNo	Obrigatório	STRING	Número da peça
TypeVar	Opcional	STRING	Variante da peça
TypeVersion	Opcional	STRING	Versão da peça
WorkingCode	Opcional	short	Código da tarefa
WorkCycleCount	Opcional	DINT	Ciclos de trabalho concluídos
CycleTimePrev	Opcional	DINT	Tempo de ciclo da última peça
OrderID	Opcional	STRING	Número da encomenda
Batch	Opcional	STRING	Número do lote da peça
MachineID	Opcional	STRING	Referência da máquina

Capítulo 5

Solução Proposta

Neste capítulo é apresentada a solução implementada para a monitorização da *Chiron* com recurso à comunicação com o *Nexeed* MES.

Depois de alguns testes e análises preliminares, verificou-se que a aquisição de dados pelo serviço DDE ou DCTL, figura 4.3, seria de uma natureza insegura, pois ambos os serviços necessitam de ser executados em plataformas que disponham do serviço DDE, que teve o seu suporte terminado no *Windows XP*. A *Microsoft* deixou de suportar este sistema operativo em 2014, o que significa que não existem *updates* de segurança para o mesmo, tornando-o extremamente vulnerável a ataques informáticos. Outra hipótese de aquisição de dados seria a instalação de sensores externos. Esta opção foi descartada inicialmente devido ao elevado investimento necessário e à dificuldade de obter resultados fiáveis dos mesmos.

Para a recolha de dados do equipamento CNC, esta solução recorre ao serviço de cliente fornecido pelos servidores OPC presente no HMI (PCU50). Os servidores OPC utilizados no PCU 50 são OPC DA e OPC AE. Estes serviços têm por base as tecnologias COM e DCOM, pelo que arquitetura da solução implementada passa pela utilização de duas aplicações desenvolvidas em *C#* na plataforma *.Net* (*Visual Studio 2017*) que são compatíveis com as tecnologias referidas anteriormente.

5.1 Arquitetura

A arquitetura implementada passa pela utilização de duas aplicações clientes dos servidores OPC DA e OPC AE presentes no HMI. A comunicação com o HMI utiliza a família de protocolos TCP/IP. Outra tarefa das aplicações é reencaminhar os dados recolhidos via telegramas XML para um servidor *Nexeed* MES.

Durante a fase de desenvolvimento das aplicações, a ligação entre o computador, onde as aplicações são executadas, e o HMI efetuou-se com recurso a uma ligação *ethernet*. Ambas as aplicações comunicam entre si através de IPC Pipes. A utilização de duas aplicações em detrimento de apenas uma deve-se ao aumento da rapidez de resolução de erros e *bugs* durante a fase de desenvolvimento.

Os telegramas são enviados para a porta do endereço IP (Internet Protocol) do servidor onde é executada a aplicação DDL do *Nexeed* MES. Depois da receção dos telegramas no DDL, os dados são tratados e o seu conteúdo é guardado numa base de dados *Oracle* presente no servidor de bases de dados. Posteriormente o servidor *Web* acede à base

de dados disponibilizando assim aplicações e serviços que permitem aos engenheiros de processo a elaboração de relatórios de qualidade, cálculo de KPIs e OEEs; aos gestores de produção o planeamento da produção e de turnos (*Shiftbook*) e à equipa de produção o acompanhamento do estado da produção, peças produzidas e a produzir, com recurso ao *Andon*. Na figura 5.1 é apresentada a arquitetura usada na solução implementada. Os módulos de *software* desenvolvidos foram duas aplicações na figura denominadas por *OPC DA Application* e *OPC AE Application*, que durante a fase de desenvolvimento foram executadas num computador portátil.

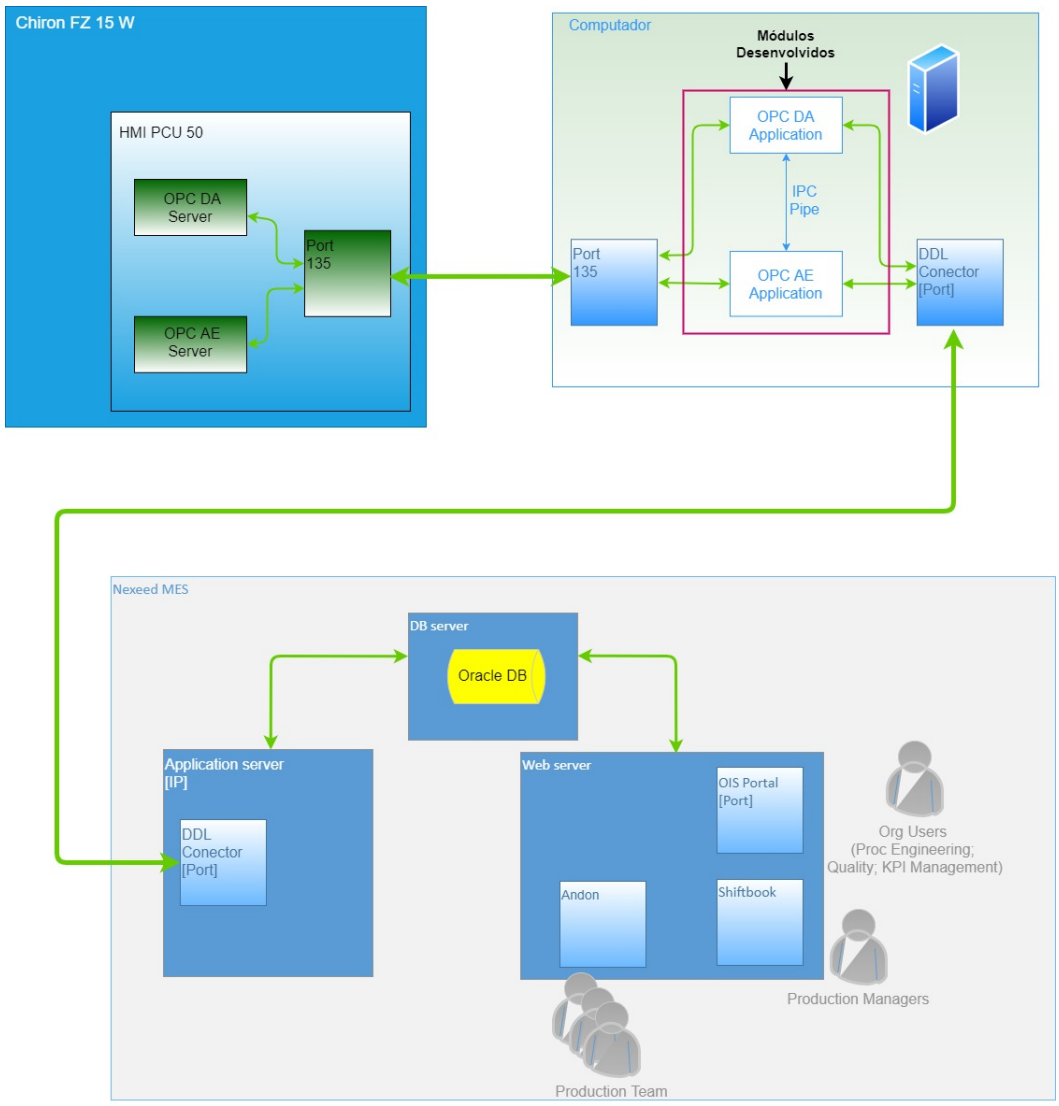


Figura 5.1: Arquitetura implementada.

5.2 Ligações OPC

Nesta secção será explicado de que forma as aplicações desenvolvidas atuam como clientes dos servidores OPC via DCE/RPC (Remote Procedure Call).

5.2.1 Ficheiro Log

Com o intuito de registar todos os erros que ocorrem durante a execução de ambas as aplicações, são criados diariamente ficheiros *.txt*, figura 5.2, onde são armazenados todos os erros e eventos de maior importância que decorreram durante o funcionamento das aplicações (envio de telegramas, sequências de conexão e erros). O nome deste ficheiro é composto pela palavra *Log* seguida pelo dia, mês e ano em que foi criado, sendo que é armazenado no mesmo diretório em que se encontra a aplicação. A função destes ficheiros é, para além da elaboração de um historial de erros, o aumento da facilidade da identificação e correção dos mesmos.

```
07-05-2020 10:28:11     Creating subscription of item/ACC/HS_PRE/$MD_MOTOR_NOMINAL_POWER failed
07-05-2020 10:28:11     Creating subscription of item/ACC/HS_PRE/$MD_MOTOR_NOMINAL_POWER_M2 failed
07-05-2020 10:28:11     Creating subscription of item/Channel/SelectedFunctions/Mval[u2,c1,#5] failed
07-05-2020 10:28:11     Creating subscription of item/PLC/Datablock/Word[c31,86,#1] failed
07-05-2020 10:28:11     Creating subscription of item/NCK/MACHINEAXIS/DRIVEPROGMESSAGES failed
07-05-2020 10:28:11     Creating subscription of item/PLC/MESSAGES failed
```

Figura 5.2: Exemplo ficheiro Log.

5.2.2 OPC DA

A aplicação responsável pela subscrição e monitorização de itens do servidor OPC DA presente no HMI PCU 50 da *Chiron* denomina-se por *opcda_ bosch*. Esta secção trata exclusivamente da aquisição de dados do servidor OPC DA.

Para a conexão como cliente e para a consequente subscrição dos itens do OPC DA, a aplicação utiliza uma biblioteca de classes, figura 5.3, que disponibilizam funções que permitem a comunicação remota com OPCs DA implementados pela *Siemens* no HMI *PCU 50 V2*.

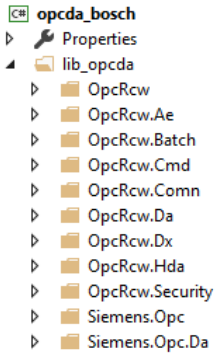


Figura 5.3: Biblioteca de classes para *Siemens* OPC DA.

Na figura 5.4 é exposto o diagrama de fluxo para o processo de conexão e subscrição de itens do servidor OPC DA.

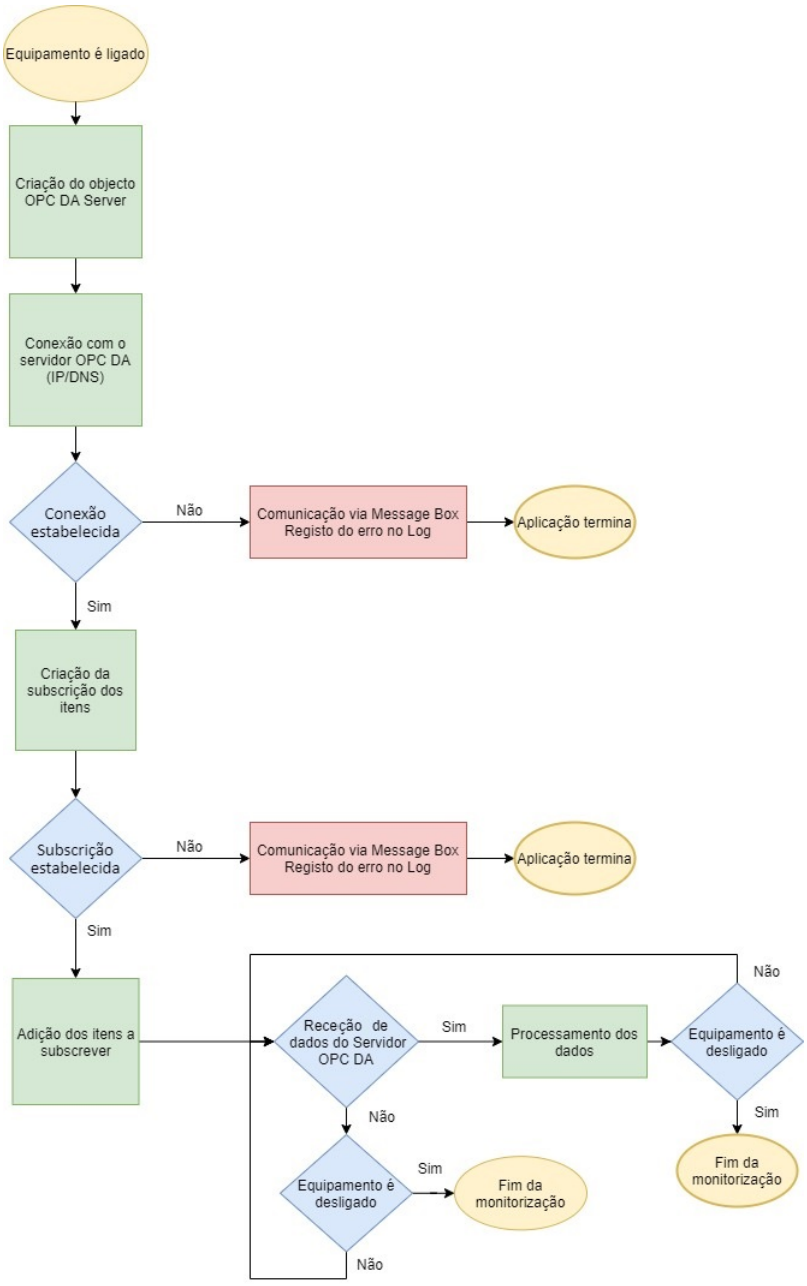


Figura 5.4: Sequência de conexão com o servidor OPC DA realizada pela aplicação *opcda_ bosch*.

O processo de conexão com o servidor OPC é iniciado pela criação do objeto *OPC DA server*, que apesar do nome é utilizado pela aplicação na sua definição como cliente. Após a criação deste objeto, a aplicação liga-se ao servidor OPC presente no equipamento com endereço IP inserido na janela da aplicação. A comunicação é feita com recurso ao protocolo DCE/RPC encapsulado na família de protocolos TCP/IP. Esta ligação com o servidor é estabelecida pela criação de um grupo público no servidor. O grupo criado pela aplicação é um contentor lógico para o cliente organizar e manipular itens. No presente caso, o grupo criado é denominado de *ReadWriteDefault*, e permite o acesso de leitura e escrita, sendo que, para a caracterização única do cliente é utilizado um GUID (Global Unique Identifier).

Logo que ocorra a criação do grupo no servidor, a aplicação já se encontra registada no mesmo como sendo um cliente e o próximo passo efetuado é a criação de uma subscrição. Esta subscrição possibilita a adição de itens ao grupo criado no servidor, sendo também nesta subscrição que é definida qual a função, *callback*, que é chamada sempre que o valor de um item é modificado e enviado pelo servidor. Sempre que um item tem o seu valor alterado, esta aplicação executa determinadas tarefas, únicas para cada item subscrito (envio de telegramas, modificação de variáveis, entre outras), que serão explicadas mais à frente.

Para a subscrição dos itens é fornecido ao servidor o nome do item e um número identificador do mesmo que será usado pelo servidor aquando da comunicação de mudança do valor do mesmo.

O endereço IP onde se encontram os servidores OPCs é introduzido numa caixa de texto da aplicação, figura 5.5, e o início da execução da mesma é definido pela seleção do botão *connect*. Após a ativação deste botão não é iniciado de imediato a conexão pois dada a natureza da aplicação, esta poderá encontrar-se alojada num local remoto sendo impossível a verificação do estado da CNC, por outras palavras se esta se encontra ligada ou desligada. Para responder a este problema a aplicação inicia uma rotina de envio de *pings* a cada cinco segundos. Caso a resposta a dois *pings* consecutivos seja bem sucedida dá-se então a rotina de conexão com o servidor. Por outro lado se dois *pings* consecutivos forem mal sucedidos, dá-se eliminação de todos os objetos relacionados com a conexão, de modo a que seja possível o estabelecimento de uma nova conexão assim que a máquina se torne a ligar.

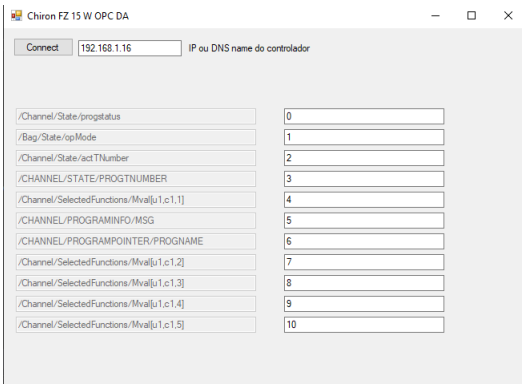


Figura 5.5: Aplicação *opcda_bosch*.

Os itens subscritos são apresentados na tabela 5.1.A seleção destes itens conjuga-se com o preenchimento e envio dos telegramas XML para o *OpCon* MES.

Tabela 5.1: Itens subscritos.

Nome	**Descrição**
/Channel/State/progstatus	Estado do programa: 1 = interrupted; 2 = stopped; 3 = in progress; 4 = waiting; 5 = aborted.
/Bag/State/opMode	Modo e operação: 0 = JOG; 1 = MDI; 2 = AUTO.
/Channel/State/actTNumber	Ferramenta atual.
/Channel/State/progTNumber	Ferramenta programada.
/CHANNEL/ProgramPointer/ProgName	Nome do programa da peça.
/Channel/SelectedFunctions/Mval[u1,c1,1]	Função M nº1 ativa.
/Channel/SelectedFunctions/Mval[u1,c1,2]	Função M nº2 ativa.
/Channel/SelectedFunctions/Mval[u1,c1,3]	Função M nº3 ativa.
/Channel/SelectedFunctions/Mval[u1,c1,4]	Função M nº4 ativa.
/Channel/SelectedFunctions/Mval[u1,c1,5]	Função M nº5 ativa

5.2.3 OPC AE

A aplicação responsável pela subscrição e monitorização de itens do servidor OPC AE presente no HMI PCU 50 da *Chiron* denomina-se por *opcae_ bosch*. Esta secção trata exclusivamente da aquisição de dados do servidor OPC AE.

Para efetuar a conexão como cliente do servidor *AE*, a aplicação utiliza uma biblioteca de classes, figura 5.6, biblioteca essa que permite a comunicação remota com o servidor OPC AE implementado pela *Siemens* no HMI *PCU 50 V2*.

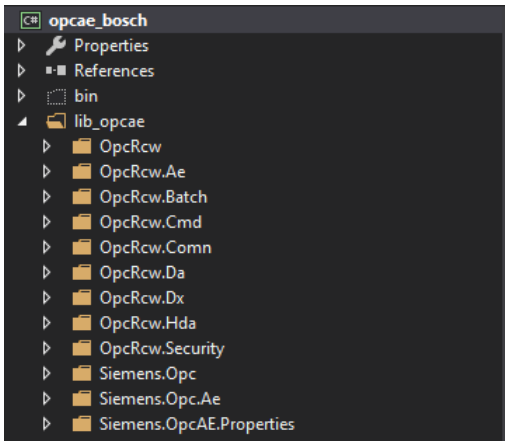


Figura 5.6: Biblioteca de classes para *Siemens* OPC AE aplicação *opcae_ bosch*

Analogamente ao OPCDA, a sequência de subscrição de eventos do servidor OPC AE está representada na figura 5.7.

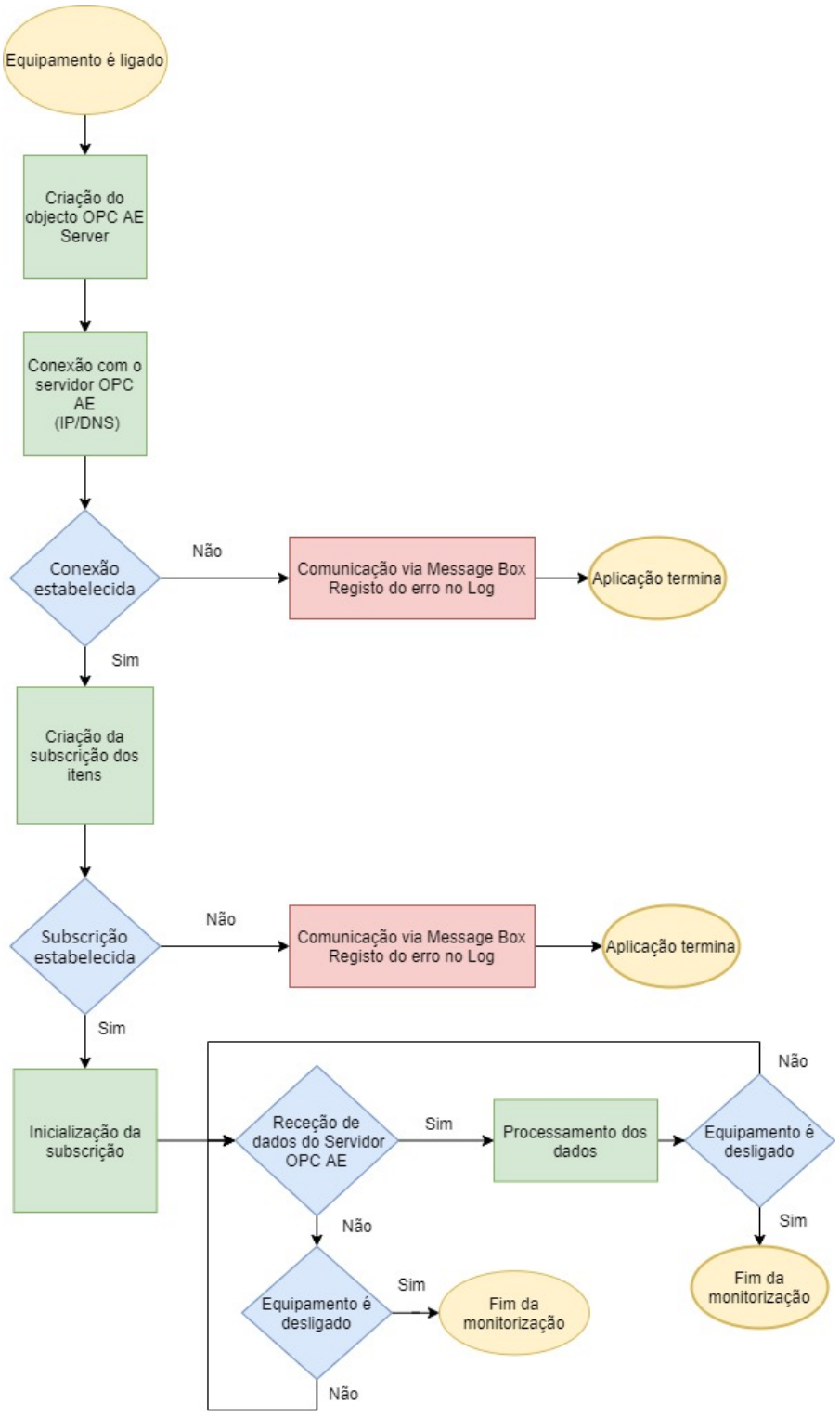


Figura 5.7: Sequência de conexão com o servidor OPC AE.

Uma vez mais, a sequência é iniciada pela criação do objeto *opc ae server*. De seguida a aplicação conecta-se ao servidor AE via TCP/IP através do protocolo DCE/RPC, presente no equipamento com endereço IP inserido na janela da aplicação *opcda_ bosch*. O passo seguinte, de acordo com as especificações do OPC AE, é a associação de uma função que é executada sempre que o servidor efetua um pedido de desconexão do cliente. Esta função desconecta a aplicação do servidor e elimina todas as interfaces de comunicação com o mesmo.

Após a conexão com o servidor é criada e configurada a subscrição de eventos. É neste passo que é definido o tamanho e o tempo do *buffer*, o estado da conexão, filtros e os tipos de eventos subscritos. Em seguida é associada a função que será chamada sempre que o servidor envia a atualização do conteúdo de um evento. A função desta *callback* é o envio de telegramas com base no conteúdo da atualização do evento. Por fim, é executado um *refresh* que representa um pedido ao servidor para que sejam enviadas todas as condições ativas e inativas e *unacknowledged* para as quais as notificações de evento correspondam aos filtros definidos. No caso desta aplicação, não são usados filtros, pois é de maior interesse receber todas as notificações de eventos. Sempre que ocorra um erro no processo de conexão ou desconexão este é guardado no ficheiro *Log* e é comunicada através de uma *message box*.

Esta aplicação não depende do seu *front-end* para funcionar, apenas disponibiliza informações acerca das atualizações de eventos recebidas, figura 5.8. Assim torna-se mais fácil no futuro uma conversão da mesma num serviço *Windows* dado que os serviços não dispõem de *front-end*.

Figura 5.8: Aplicação *opcae_ bosch*.

5.2.4 Comunicação com Nexeed MES

Após a recolha e tratamento de dados, ambas as aplicações necessitam de comunicar com o *Nexeed* MES. A comunicação é feita com recurso a telegramas XML, conforme o exposto na subsecção 4.3.4. Para a comunicação com o DDL, e para o preenchimento e envio dos telegramas, ambas aplicações recorrem a um SDK (Software Development Kit). O SDK é composto por DLLs (Dynamic Link Library) que quando integrados no projeto disponibilizam funções de ligação ao servidor MES e envio estruturado de telegramas. Com recurso a ficheiros XML e XSD (XML Schema Definition) é configurada a estrutura dos telegramas a enviar. Este SDK foi disponibilizado pela *Bosch Termotecnologia*. É ainda importante referir que o SDK em questão é utilizado na empresa em todas as comunicações com o *Nexeed* provenientes de aplicações programadas em *C#*. Para o funcionamento de ambas as aplicações é necessário que os ficheiros, à exceção do *Client.Config.xml*, descritos na figura 5.9, se encontrem na mesma pasta que os executáveis.

OpCon e *Nexeed* são o mesmo sistema MES. Inicialmente, a solução MES desenvolvida pela *Bosch* para implementação em fabricas do grupo *Bosch* designava-se por *OpCon*, tendo este mudado para *Nexeed* quando a *Bosch* decidiu comercializar este produto para empresas fora do grupo.

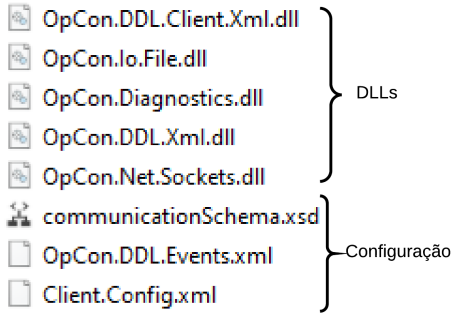


Figura 5.9: ATMO XML SDK.

Os ficheiros *comunicationSchema.xsd* e *OpCon.DDL.Events.xml* definem os atributos obrigatórios e opcionais elemento `header`, do sub elemento `location` e do `res head` a serem enviados nos telegramas XML. Todos os telegramas que podem ser enviados com recurso ao SDK, tabela A.1, têm o seu elemento `event` definido por estes dois ficheiros, quer nos atributos obrigatórios e opcionais, como também no tamanho mínimo e máximo dos mesmos.

Client.config.xml

O ficheiro *Client.Config.xml* tem a função de configurar todo o processo de comunicação com o *Nexeed*. Como se trata de um ficheiro que poderá ser alterado ao longo do funcionamento das aplicações, este é compartilhado pelos programas *opcda_ bosch* e *opcae_ bosch*, sendo assim, apenas é necessário modificar as configurações num ficheiro só, eliminando a existência de duplicados. A sua localização é o diretório da aplicação

opcda_bosch O código presente na lista de código 5.1 representa todo o conteúdo do ficheiro *Client.Config.xml* usado pelas aplicações deste projeto.

Código Fonte 5.1: Ficheiro Client.Config.xml

```
<?xml version="1.0" encoding="utf-8"?>
<DDL.Client.Config xmlns="Bosch.OpCon.DDL.Client.Xml">
  <Logging>
    <LogLevel> Info </LogLevel>
    <LogFile> .\Logfile.tx5 </LogFile>
    <MaxSize> 55 </MaxSize>
    <MaxAge> 3 </MaxAge>
    <MaxLogFiles> 3 </MaxLogFiles>
    <TelegramLogDir> .\TelegramLog\</TelegramLogDir>
  </Logging>
  <Validation>
    <Enabled> true </Enabled>
    <SchemaFile>.\communicationSchema.xsd</SchemaFile>
  </Validation>
  <Locations>
    <Location source="stat-001" lineNo="851" statNo="185" statIdx="1"
    fuNo="1" workPos="1" toolPos="1" processNo="1000" processName="
    ChironTeste" application="CNC" />
  </Locations>
  <OpConXml>
    <ContentType> 1 </ContentType>
    <SetTimeStamp> false </SetTimeStamp>
  </OpConXml>
  <Communication>
    <Service name="localhost">
      <Host> 192.168.1.68 </Host>
      <Port> 55065 </Port>
      <RetryCount> 3 </RetryCount>
      <RetryDelay> 1000 </RetryDelay>
      <Timeout> 5000 </Timeout>
    </Service>
  </Communication>
</DDL.Client.Config>
```

O elemento `<Logging>` é utilizado na configuração de um ficheiro onde serão guardadas todas as ações e erros realizados no processo de comunicação com o *Nexxed MES*, erros de ligação com o servidor ou falhas nos envios dos telegramas. Os seus sub-elementos permitem configurar o nível de dados a armazenar (`<LogLevel>`), configurar o nome do ficheiro e a localização do ficheiro (`<LogFile>`), o seu tamanho máximo (`<MaxSize`), a duração máxima em dias que poderá ser utilizado (`<MaxAge>`), o número máximo de ficheiros passíveis de existir (`<MaxLogFiles>`) e ainda uma pasta onde serão armazenadas cópias de todos os telegramas enviados(`<TelegramLogDir>`).

O elemento `<Validation>` tem como sub-elementos o `<Enabled>` que ativa ou desativa a validação dos telegramas enviados e recebidos com base no ficheiro que descreve a estrutura aceite para os telegramas. O diretório deste ficheiro é especificado no sub-elemento `<SchemaFile>`

No elemento `<Locations>` encontram-se os sub-elementos `<Location>` . Neste sub-elemento está descrita a localização da *Chiron* no chão de fábrica de acordo com os atributos da tabela 4.4. Apesar de este equipamento não se encontrar ligado diretamente

ao *Nexeed* MES, a sua localização já se encontrava definida pela planta fabril. No caso de um programa enviar telegramas de várias localizações é possível especificar vários elementos `<location>`, sendo que são endereçados tendo por base o seu item `<source>`. As propriedades da ligação ao servidor MES são definidas no elemento `<Comunication>`. À semelhança do elemento `<Locations>` é possível definir várias ligações dentro de sub-elementos `<Service>`. Os atributos definidos para cada comunicação são: nome da ligação `<name>`, o endereço IP do servidor `<host>`, a porta onde o servidor atua `<Port>`, o número de tentativas de ligação `<RetryCount>`, o intervalo de tempo entre tentativas `<RetryDelay>`, e o intervalo de tempo entre a última tentativa e o cancelamento completo da ligação `<Timeout>`. A definição destes parâmetros teve por base a utilização de um simulador de servidor recetor de telegramas do MES executado no endereço *192.168.1.68*.

Processo de Envio dos Telegramas

Para que seja possível o envio dos telegramas XML para o *DDL Conector*, ambas as aplicações durante a sua inicialização estabelecem uma conexão TCP/IP com um programa que está a correr no endereço IP e na porta especificados pelos atributos do subelemento do elemento `source` do ficheiro *client.config.xml*. Após estabelecida a conexão, torna-se possível o envio de telegramas contendo informações sobre o funcionamento da CNC para o *Nexeed MES*. Quando as aplicações são fechadas, a conexão é terminada e a livraria é encerrada. Mais uma vez, é utilizado o ficheiro *Log* para o armazenamento de todos os erros decorrente do processo de conexão e desconexão. Na figura 5.10 está representada a sequência de conexão com um *DDL Conector* que ambas as aplicações executam durante a sua inicialização.

Dada a inicialização da livraria e estabelecida a conexão com o *DDl conector* do *Nexeed MES*, as aplicações ficam aptas a enviar os telegramas para o mesmo. Para qualquer telegrama enviado, a sequência de preenchimento e envio é semelhante, apenas mudando os atributos do `Event` e a inclusão de elementos no `body`. Esta sequência ocorre de acordo com a sequência da figura 5.11. O primeiro passo executado é a criação do objeto que é responsável pelo controlo e preenchimento do telegrama. Este objeto quando é criado é associado à conexão estabelecida. De seguida são preenchidos os atributos `EventSwitch` e `ContentType` e é também definido se a aplicação está à espera de uma resposta do servidor. Para os telegramas enviados pelas aplicações deste projeto, este campo foi definido como falso, pois não existe a necessidade de uma resposta do servidor. Todos os restantes atributos do `header` à exceção do `EventName` são preenchidos automaticamente pelas funções do SDK.

O passo seguinte é o preenchimento do subelemento `Location` com recurso ao item `source` de um dos sub-elementos do elemento `Location` presentes no ficheiro *client.config.xml*. Desta maneira não é necessário preencher manualmente este subelemento e facilmente é possível que uma só aplicação envie telegramas de processos com diferentes localizações. Seguidamente, dá-se o preenchimento do `Reshead` e a inclusão dos elementos configuráveis do `body`. Por fim é definido qual o tipo de telegrama a enviar e consequentemente dá-se o preenchimento do `Event` e do `ContentType` pelo SDK. Seguidamente, é enviado o telegrama. Se ocorrer um erro no envio ou o preenchimento dos campos do telegrama esse será registado no ficheiro *Log* e também no ficheiro definido no `<LogLevel>` do *client.config.xml*.

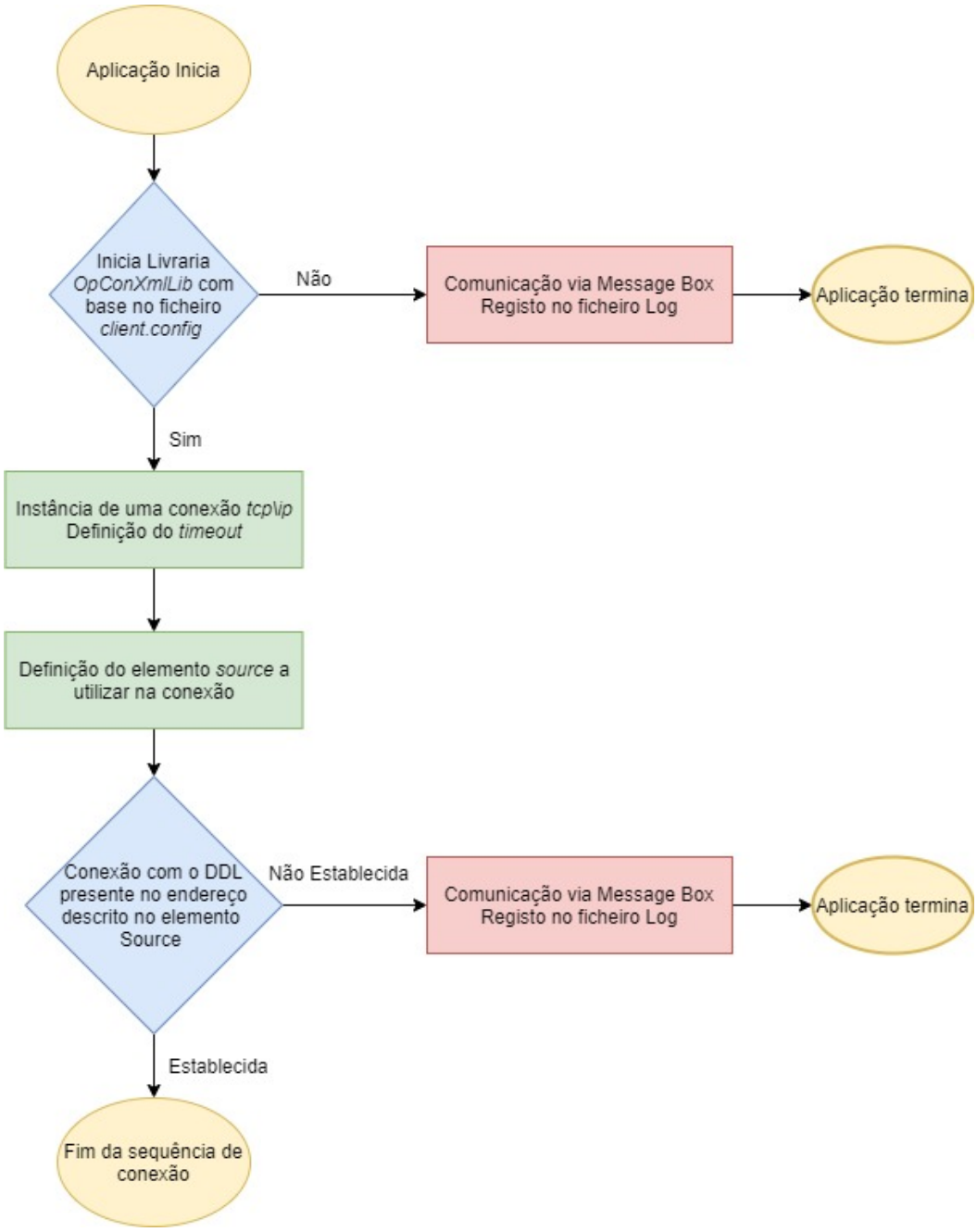


Figura 5.10: Sequência de conexão com o *DDL Conector*.

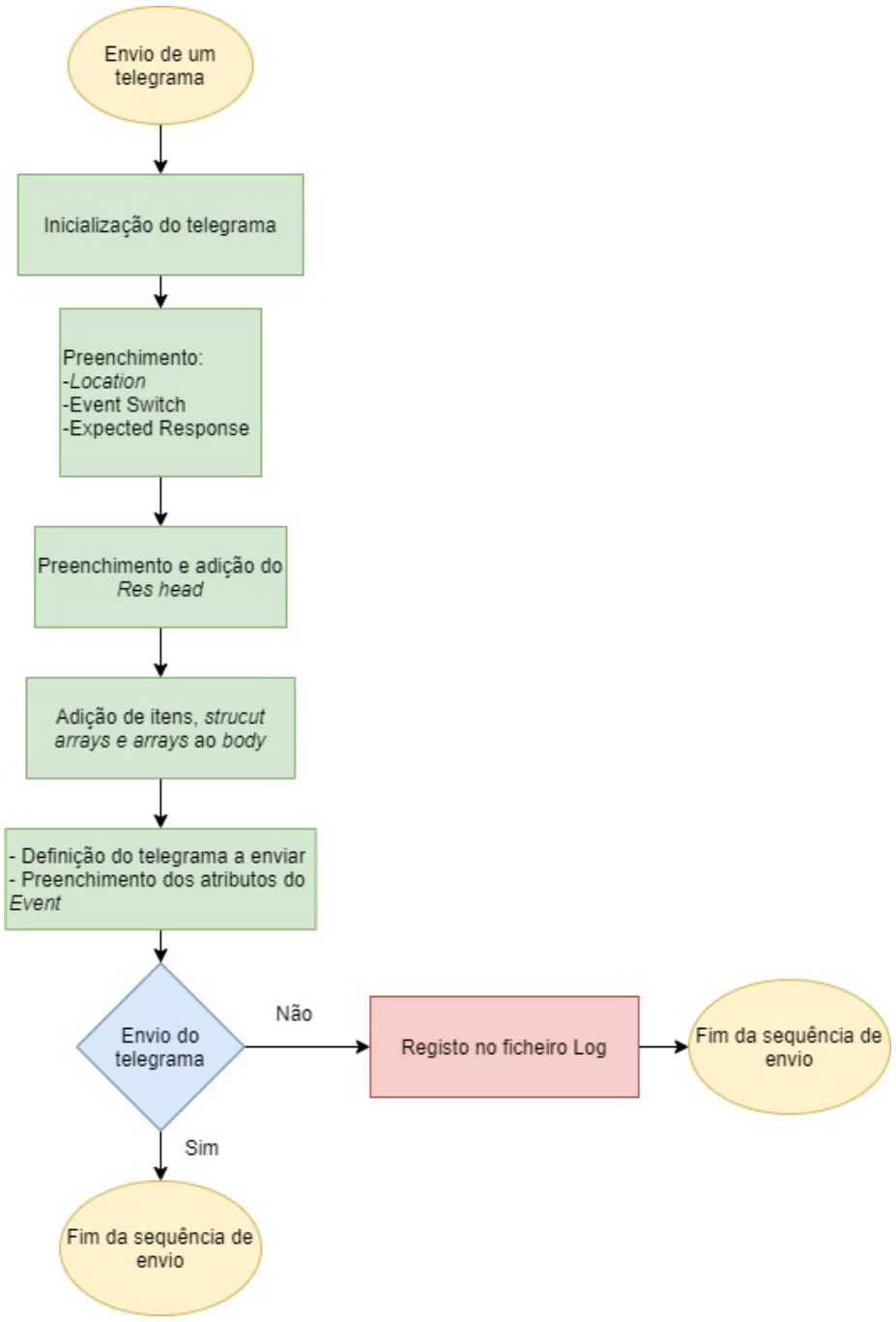


Figura 5.11: Sequência de preenchimento e envio de telegramas.

5.2.5 Comunicação entre Aplicações

Para a partilha de informações entre ambas as aplicações, estas recorrem a *IPC Pipes*. Esta é uma das muitas soluções utilizadas por aplicações *Windows* para a partilha de informação, sendo que é a mais fácil e simples de utilizar dada a simplicidade das informações partilhadas pelas aplicações deste projeto. Deste modo, durante a inicialização são criados dois *pipes* e cada aplicação escreve no *pipe* criado pela outra aplicação e escuta o que é escrito no *pipe* por si criado. Sempre que uma aplicação envia informação pelo *pipe*, a outra aplicação recebe essa informação numa função *callback*, sendo esta processada pela mesma função.

As mensagens trocadas entre ambas as aplicações encontram-se representadas na figura 5.12. Todas as mensagens entre aplicações são iniciadas pelo carácter '.' e terminadas pelo caractere '-'. Desta forma é fácil identificar o início e o fim das mensagens. Quando a aplicação *opcae_bosch* inicia, esta escreve no seu *pipe* a mensagem *'.getip-'* para a aplicação *opcda_bosch* que lhe responde com o *IP* da CNC. Após a receção do endereço *IP*, a aplicação *opcae_bosch* envia a mensagem *'.getcc-'* para a qual a aplicação *opcda_bosch* responde com a localização do ficheiro *client.config*. Se não existir resposta às mensagens enviadas num período de 10 segundos, será exibida uma mensagem a informar da não existência de resposta e é registado um erro de comunicação no ficheiro *LogFiles*.

Após a receção da localização do ficheiro *client.config*, esta aplicação tenta ligar-se ao servidor DDL com a configuração presente no seu elemento *Service* com o nome de *localhost*. Ambas as aplicação usam as configurações da *Location* do sub-elemento *source* designado por *stat-001*. Sempre que a CNC desliga e liga, a aplicação *opcda_bosch* envia as mensagens *'.conect-'* e *'.disconnect-'*. Ao receber estas mensagens, a aplicação *opcae_bosch* inicia ou termina a sua conexão com o servidor OPC.

Como irá ser apresentado mais à frente neste documento, um dos itens preenchido no telegrama de erro enviado pela aplicação *opcae_bosch* é o modo de operação. Para recolher esta informação que se encontra disponível pelo OPC DA, a aplicação escreve no *pipe* a mensagem '.getopmode-', sempre que envia um telegrama. A resposta desta pergunta é enviada para o *pipe* criado pela aplicação com o seguinte formato '.modeis*mode*' seguida pelo modo como exemplo,'.modeis1-'.

Se ocorrerem erros na criação dos *pipes*, ou nos envios de mensagens, estes serão comunicados via *textBox* e registados no ficheiro *LogFile*.

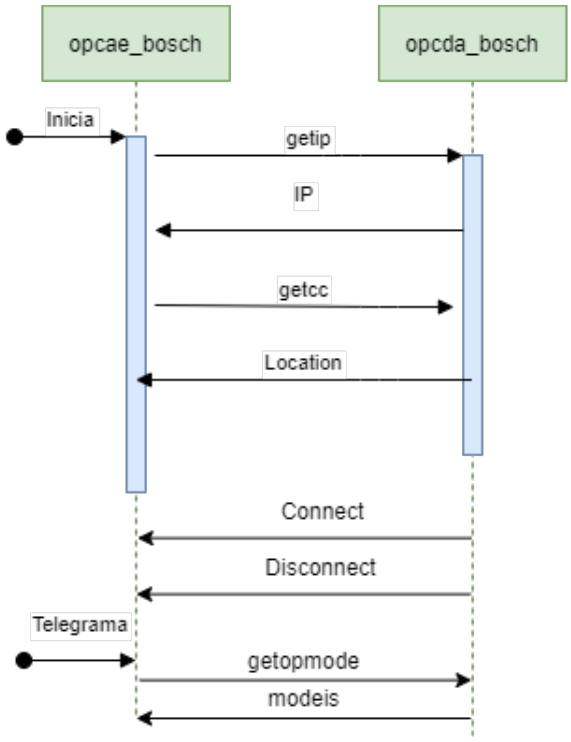


Figura 5.12: Sequência de preenchimento e envio de telegramas.

5.3 Setup

Para que as ambas as aplicações funcionem de forma correta é necessário modificar algumas configurações do controlador (PCU 50) e do servidor onde as aplicações serão executadas.

Os passos necessários para as alterações de configuração do HMI estão descritas no anexo C.1. Estas alterações passam pela desativação do bloqueio de comunicações TCP/IP pela Firewall para os servidores OPC presentes no HMI e do bloqueio da porta TCP 135 usada pelo serviço RPC [45].

Ambas as aplicações **opcae_bosch** e **opcda_bosch**, assim como a porta 135 têm necessariamente que ser adicionados às listas de exceções da Firewall do equipamento onde serão executadas. Para o correto funcionamento do serviço DCE/RPC é obrigatória uma autenticação de credenciais. Esta autenticação de segurança é realizada através do nome de utilizador e da *passsword*. Dada esta condição o equipamento onde as aplicações são executadas necessita de possuir um utilizador com o mesmo nome e *passsword* que o HMI. No presente caso o *Windows XP* do PCU50 possui um utilizador de nome "*auduser*"protegido com a palavra passe *"SUNRISE"*.

Como requisito da *Bosch Termotecnologias* esta *password* não pode ser alterada. Dada esta condição, o computador utilizado para a execução das aplicações foi alvo da criação de um novo utilizador com nome e *password* iguais aos do HMI. O computador utilizado para a execução das aplicações possui o sistema operativo *Windows 10*. Este sistema operativo não permite, por defeito, a utilização de *passwords* tão simples como *SUNRISE* pelo que os passos para a desativação das regras que definem o nível

de complexidade das *passwords* foram alteradas segundo os passos definidos no anexo C.2.1. A redução do nível de complexidade da *password* tem como consequência uma diminuição da segurança ,quer do equipamento onde são executadas as aplicações, quer do HMI, facilitando assim o acesso indevido ao equipamentos por utilizadores da rede interna. Contudo esta redução na segurança não é crítica na fase de testes, pois ambos os equipamentos não se encontravam conectados à rede interna da fabrica. Aquando da implementação das aplicações num regime produtivo, a *password* do equipamento será alterada para conter um maior nível de complexidade e consequente aumento da segurança.

É também necessário instalar *OPC Core components* no equipamento onde as aplicações iram ser executadas.

A fim de existir uma ligação TCP/IP entre o servidor e o HMI é necessário que ambos possuam as mesmas configurações de rede. Dado que o HMI possui endereço IP estático, 192.168.1.16, e uma máscara de rede 255.255.255.0, o computador das aplicações necessita de conter um endereço IP igual a este com uma alteração nos últimos dois dígitos. Para isto é necessário desativar a utilização protocolo DHCP (Dynamic Host Configuration Protocol) caso este se encontre ativo, seguindo o exemplo apresentado no anexo C.2.2. Este protocolo através de um servidor é capaz de atribuir automaticamente diferentes endereços IP para todos os computadores de uma rede consoante eles solicitam a conexão de rede. Como no presente caso nenhum dos dois equipamentos, que fazem parte da rede, possui um servidor DHCP, foi necessário atribuir manualmente um endereço IP ao computador onde as aplicações são executadas.

5.4 Envio e Gestão dos Telegramas

Nesta secção serão analisados os resultados do funcionamento das aplicações. Todos os telegramas enviados por ambas as aplicações encontram-se representados na figura 5.13. Os telegramas *plcError* são enviados pela aplicação *opcae_bosch* sendo que os restantes são da responsabilidade da aplicação *opcda_bosch*. Os telegramas *PartProcessingPaused* e *PartProcessingAborted* só ocorrem em condições extraordinárias, para as quais o programa é interrompido ou abortado durante a maquinação. Já todos os outros telegramas são enviados em condições normais de funcionamento do equipamento.

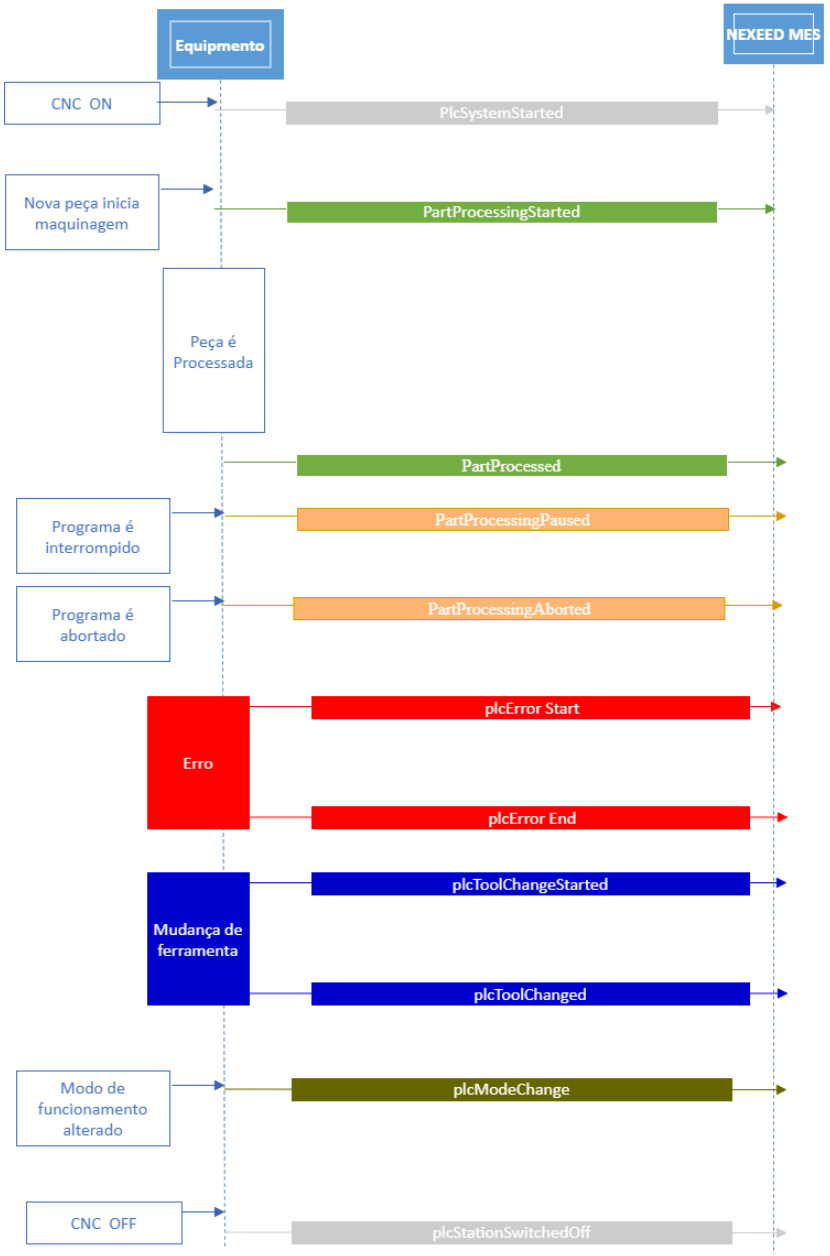


Figura 5.13: Telegramas enviados.

5.4.1 PlcSystemStarted e PlcStationSwitchedOff

Nesta estação o operador liga a corrente da CNC no início do turno e desliga a mesma no fim do turno. Dada esta situação, a forma mais simples de se ter conhecimento se o equipamento está ligado ou desligado foi o recurso a resposta do envio de um *ping*. Após o seu início, a aplicação *opcda_bosch* envia a cada cinco segundos um *ping* para o endereço IP da CNC. A partir do momento em que o equipamento inicia, este responde ao envio dos *pings*. Se dois *pings* consecutivos obtiverem uma reposta é iniciada a monitorização dos itens *OPC DA*, a aplicação envia pelo *IPC Pipe* uma mensagem para a outra aplicação iniciar a monitorização dos itens *OPC AE* e é enviado um telegrama *PlcSystemStarted*, anexo D.2. No caso de repentinamente deixar de existir respostas para dois *pings* consecutivos é destruída a ligação com o *OPC DA*. A aplicação *opcae_bosch* é informada de que deve destruir a conexão com o servidor *OPC AE*, por fim é enviado um telegrama com o `EventName` *pPlcStationSwitchedOff*, anexo D.3.

Ambos os telegramas são simples, apenas possuindo o `header` preenchido e o `event`. A informação mais importante contida nestes telegramas é a localização do equipamento e o *timeStamp*. Com esta informação o *Nexeed MES* possui todo o que necessita para saber qual o período de tempo em que a máquina esteve ligada, sendo esta informação importante para o cálculo do OEE. A escolha de 2 *pings* consecutivos para avaliar a ligação do equipamento foi arbitrário. Por um lado a utilização de apenas 1 *ping* poderá culminar no envio errado de telegramas devido a falhas momentâneas na rede, por outro a necessidade de mais *pings* com ou sem resposta atrasa o envio de telegramas face ao estado real do equipamento. Este método possui a vantagem de não ser necessária a utilização de componentes adicionais para a identificação do momento em que o equipamento é ligado e desligado. Como desvantagem advém o facto do envio errado de telegramas na eventualidade de falhas na rede *ethernet*.

5.4.2 PlcToolChangeStarted e PlcToolChanged

O item */Channel/State/progTNumber* do *OPC DA* muda de valor sempre que na leitura do programa peça é encontrado uma função *M06*, que diz respeito à troca automática de ferramenta. O valor deste item corresponde à posição da ferramenta no carrossel porta ferramentas da máquina. Logo que este item seja alterado é enviado um telegrama com o `EventName` *plcToolChangeStarted* , anexo D.8 em que o item `identifier` do `event` corresponde ao número da posição da ferramenta. Do mesmo modo sempre que se dá o fim do processo de troca de ferramenta item */Channel/State/actTNumber* muda o seu valor para a posição da ferramenta em uso e a aplicação envia um telegrama *plcToolChange*, anexo D.9 análogo ao telegrama anterior. Este método de identificação de ferramentas pela sua posição só é possível pois a CNC possui um carrossel que pode alojar até 48 ferramentas e apesar de serem maquinados muitos modelos diferentes de peças, as ferramentas são utilizadas em diversos modelos e não é necessário a localização das ferramentas no carrossel. Com esta condição é possível saber as características de uma ferramenta apenas sabendo qual é a sua posição no porta ferramentas. Com o envio de ambos telegramas torna-se possível identificar os tempos de troca de ferramentas, e os tempos de utilização da mesma.

Atualmente o operador da máquina é responsável por verificar, no início do turno e em intervalos de peças definidos pelo engenheiro de processo, com recurso a equipamentos de metrologia se as medidas dos elementos maquinados se encontram dentro dos limites

estabelecidos. É com esta verificação que é detetado o desgaste das ferramentas. Apesar de não ser uma métrica correta de se usar, o tempo de utilização das ferramentas pode ser utilizado para uma manutenção preventiva que é melhor que a situação atual onde é necessário ter uma peça mal maquinada para se efetuar uma mudança de ferramenta. Se o intervalo de peças entre verificações for elevado, pode ocorrer que um elevado número de peças seja rejeitado e vá para a sucata, antes que seja identificada a necessidade de trocar ou retificar uma ferramenta.

5.4.3 PlcModeChange

O comando numérico *Siemens 840 D* possui 3 modos de operação MDI (Manual Data Input), que permite a escrita de código ISO diretamente na consola HMI; modo *JOG* onde é possível o movimento manual dos eixos e por fim o modo automático que é ativo sempre que um programa peça é lido e executado. O item OPC que contém a informação sobre o modo de funcionamento em uso é */Bag/State/opMode* e sempre que este item muda de valor, a aplicação envia um telegrama com o `EventName` de *plcModeChange*, anexo D.11. No elemento `Event` apenas são preenchidos dois itens dos 8 possíveis, sendo estes o `modeOn` com o valor de *True* que indica que o modo está ativo e o `operationMode` que é o número do modo em operação (0 JOG, 1 MDI, 2 AUTO). Para uma melhor compreensão do modo de funcionamento ativo é adicionado no `body` um item que descreve o modo de funcionamento.

A informação disponibilizada por este telegrama é fundamental para o cálculo de OEE e KPIs uma vez que o tempo de funcionamento em que a máquina pode efetivamente produzir é igual ao tempo com o modo de funcionamento em modo automático.

5.4.4 PartNumber

Como referido anteriormente, um dos itens subscritos pelo programa *opcda_bosch* é o nome do programa peça em execução. Sempre que o programa inicia ou que se dá uma mudança no programa peça, é executada um rotina que ao nome do programa associa o identificador das peças produzidas e a quantidade de peças produzidas por ciclo. Esta associação é feita com recurso a um ficheiro XML que se encontra presente no mesmo diretório do programa. A configuração deste ficheiro encontra-se apresentada no anexo D.1. A utilização deste tipo de ficheiros como pequena base de dados local é eficiente e simples quando comparada com outras soluções, tanto no que diz respeito ao acesso à informação, como à alteração e adição de novos elementos.

Cada um dos sub elementos `<part>` deste ficheiro é constituído por três itens. O item `<progname>` é como o próprio nome indica, o nome do programa peça que será utilizado pelo *opcda_bosch* na pesquisa dos outros itens. O *partidentifier* é utilizado no preenchimento do item *identifier* do elemento `event` dos telegramas enviados que se relacionam com as peças produzidas. A quantidade de telegramas *Part Processed*, *Part Processing Started*, *Part Processing Paused* e *Part Processing Aborted* é definida pelo item `<partspercycle>`.

5.4.5 PartProcessingPaused e PartProcessingAborted

O estado de execução do programa peça é comunicado pelo item *progstatus*, 5.1, sempre que o estado de execução do programa de maquinagem passa de *running* para *in-*

terrupted ou *stopped* ou *waiting* é enviado conjunto de telegramas *PartProcessingPaused*, anexo D.5, com base no número de peças maquinadas por programa. Caso este passe de *running* para *aborted*, são enviados telegramas *PartProcessedAborted*, anexo D.6. Mais uma vez o número de telegramas enviados depende do número de peças maquinadas em cada mesa que esteja registado no ficheiro *partnumber.xml*.

5.4.6 PlcError

A gestão do envio de erros é feita exclusivamente pelo programa *opcae_ bosch*. Após a subscrição do servidor OPC AE por parte da aplicação, esta é notificada sempre que um evento de condição tem início ou fim. Todos os dados relativos aos alarmes do equipamento são comunicados ao *Nexeed* através de um telegrama *PlcError*, anexo D.10. Como visto anteriormente o servidor *OPC AE* disponibiliza variadas informações sobre cada condição. Para o presente trabalho apenas foram utilizados as informações que apresentavam maior relevância para o *Nexeed*. A tabela 5.2 apresenta uma relação entre as características recolhidas e os itens enviados no telegrama.

Tabela 5.2: Relação característica das condições com itens enviados.

OPC Client	**Telegrama**
Name	*errorNo*
Name	*errorText*
Tipo de Evento	*errorType*
Tipo de Evento	*modeOn*
Tipo de Evento	*EventCategory*
Severidade	*Severity*
NewSate	*errorState*
Cookie	*Cookie*

Os eventos monitorizados pelo OPC AE do equipamento contêm o *Name*, constituído pelo número do evento e pela descrição, sendo que a descrição se encontra entre parênteses. Deste modo a aplicação *opcae_ bosch* separa a *string* em dois e utiliza os campos da separação para preencher o `errorNo` e o `errorText`.

Os tipos de Eventos que constituem este servidor podem ser do tipo, *Simple, Condition, Tracking*. O item `EventCategory` é uma string que contém o tipo de evento. Já o item `errorType` pode ter o seu valor igual a 1 se o evento for um erro, 2 se for um aviso e 3 se for uma informação. Para o *Nexeed*, o item `modeOn` indica se os erros deverão ser inseridos como paragem não planeada no *Shiftbook*. Eventos do tipo *Condition* são enviados com `modeOn` *True* e `errorType` igual a 1 e todos os outros tipos de eventos têm os seus telegramas preenchidos com com `modeOn` *False* e `errorType` igual a 3.

Para o *Nexeed*, um *plcError* com *errorState* igual a 0 marca o início do erro. Já no caso deste campo tomar o valor de 1 é marcado o final do mesmo. Para o preenchimento deste item é utilizado o atributo *Newstate* que quando tem o valor igual a 5 marca o início da condição e o valor igual a 7 marca o fim da mesma.

O item *Severity* e *Cookie* são preenchidos com base na severidade e na *Cookie* associadas ao evento. Estes dados são fornecidos pelo servidor OPC AE sempre que este comunica o início ou fim de uma condição.

Como referido anteriormente, o preenchimento do item `OperationMode` advém da comunicação com a aplicação *opcda_ bosch* e pode ser 0 se o modo for JOG, 1 se o modo for MDI ou ainda 2 se o modo for automático.

Todos os outros campos do elemento `event` são opcionais pelo que atualmente são enviados com um valor nulo.

5.4.7 PartProcessingStarted e PartProcessed

No funcionamento normal de um equipamento CNC, o fim da maquinação de uma peça é marcado no seu código G pela função M30. No presente caso, a partir do momento em que o operador seleciona um programa peça este corre em contínuo sem nunca terminar, figura 5.14. Com este tipo de programa peça não é necessário que o operador utilize o HMI do equipamento para iniciar o ciclo de maquinação, apenas necessitando de colocar as peças a maquinar, retirara as peças maquinadas e pressionar o botão de confirmação de início de ciclo. Deste modo torna-se difícil identificar o momento em que um ciclo de maquinação inicia e termina. De um modo geral todas os programas peças contêm diferentes subprogramas de maquinação, de tal modo o fim de um subprograma torna-se também inválido para a deteção do fim de um ciclo de maquinação. De modo a resolver esta dificuldade, o programa *opcda_ bosch* identifica o início e o fim de um ciclo de maquinação completo para uma mesa de trabalho pela identificação da função M771, que quando chamada concretiza uma rotação das mesas de trabalho. Esta função é comum a todos os programas peça presentes no equipamento e é fundamental para a troca de mesas.

Como referido anteriormente a aplicação *opcda_ bosch* subscreve todos os itens que se referem as funções M ativas. Sempre que qualquer um destes itens toma o valor de 771, o programa envia vários telegramas *PartProcessed*, anexo D.7. O número de telegramas enviados e o identificador das peças produzidas dependem dos itens associados ao nome do programa em execução no ficheiro *PartNumber*.

De igual forma, sempre que a função M771 deixa de estar ativa, quer dizer que ocorreu a troca de mesa e são enviados telegramas *PartProcessingStarted*, anexo D.4.

A ativação da função M771 nem sempre culmina no envio dos telegramas *PartProcessed*, no caso de ser a primeira maquinação, as mesas têm que rodar para que as peças a maquinar se encontrem na posição necessária, sendo que para esta, e só esta, ocorrência nenhum telegrama é enviado. A desativação desta função continua a causar envio de telegramas *PartProcessing Started*. A primeira maquinação é identificada quando é ativada pela primeira vez a função M771, após o equipamento entrar em ciclo automático.

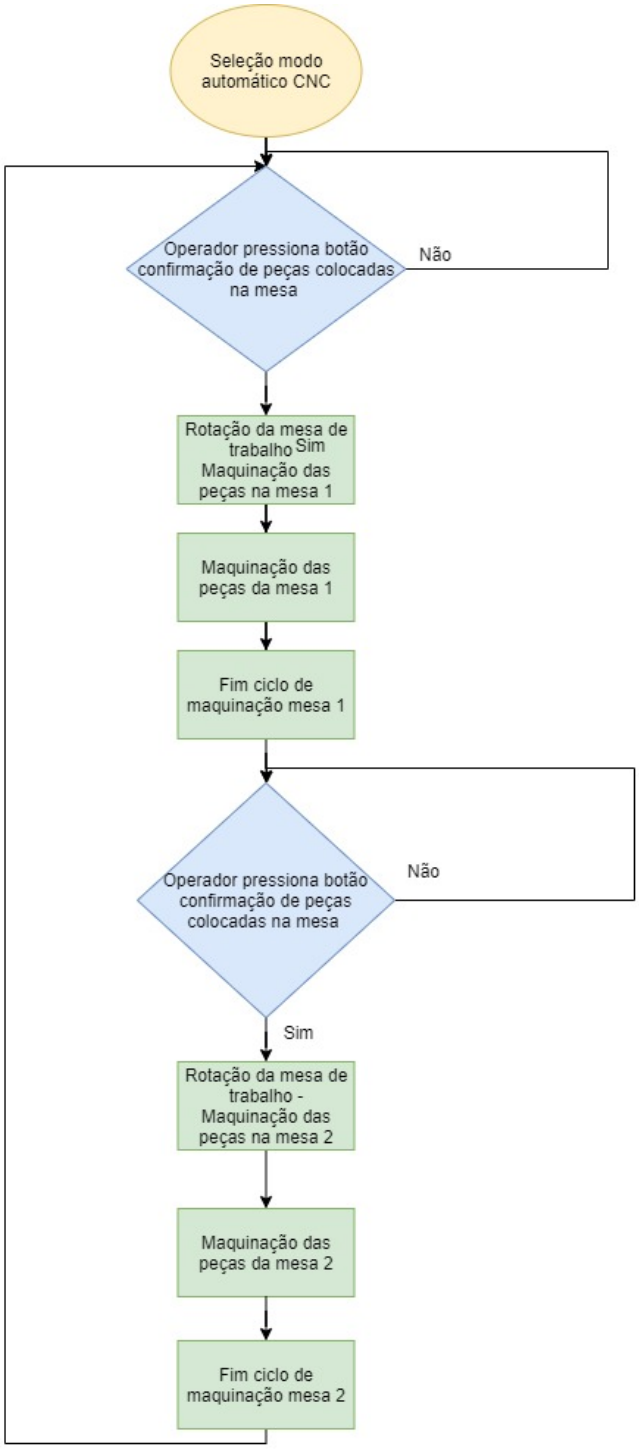


Figura 5.14: Sequência de execução de programas peça ciclo automático.

Capítulo 6

Análise de Resultados

Neste capítulo pretende-se analisar a qualidade do sistema criado e avaliar o resultado da sua implementação.

O objetivo do presente projeto é a criação de uma solução com a capacidade de recolher dados de um comando numérico *Siemens* e que fosse capaz também de enviar os dados recolhidos para o sistema MES implementado na empresa *Bosch Termotecnologia*.

A arquitetura implementada deu uma resposta eficaz aos objetivos propostos, de uma forma rápida e barata. O desenvolvimento das aplicações em *.net* mostrou-se vantajoso na implementação das bibliotecas de comunicação, quer com o equipamento, quer com o *Nexeed MES*. A criação da subscrição de itens nos servidores OPCDA e OPCAE, requer a indicação da taxa de atualização solicitada. Esta taxa define o intervalo de tempo entre atualizações para o mesmo item. Se este item variar várias vezes neste intervalo, apenas é comunicado o seu valor em ciclos com uma duração igual à fornecida. Para ambas as subscrições este intervalo é definido como 0 segundos. Após a criação da subscrição os servidores atribuem o menor intervalo de tempo por si suportado para a taxa de atualização e enviam esta informação para as aplicações cliente. Para os servidores OPC deste HMI, este intervalo de tempo é de 100ms, o que significa que pode existir um atraso máximo de 100ms entre a alteração do valor de um item e a sua comunicação às aplicações clientes Open Platform Communications. Este valor é perfeitamente aceitável para as funções destas aplicações.

Devido à situação de pandemia que se vive atualmente no nosso planeta, infligida pelo vírus da *Covid-19*, e também devido às políticas de privacidade e de segurança de dados implementadas pelo grupo BOSCH, não foi possível testar o envio dos telegramas diretamente para o Nexeed MES.

Dada a impossibilidade do envio dos telegramas para o MES foi utilizado um programa de testes fornecido pela *Bosch* que atua como um *DDL Connector* local. Este programa denomina-se de *CommTester* e atua como um servidor que escuta a porta TCP/IP 55065 sendo que recebe a armazena todos os telegramas XML enviados para a respetiva porta. O teste das aplicações desenvolvidas foi realizado recorrendo a computador portátil conectado ao HMI da CNC e o programa *Commtester* foi executado no mesmo computador.

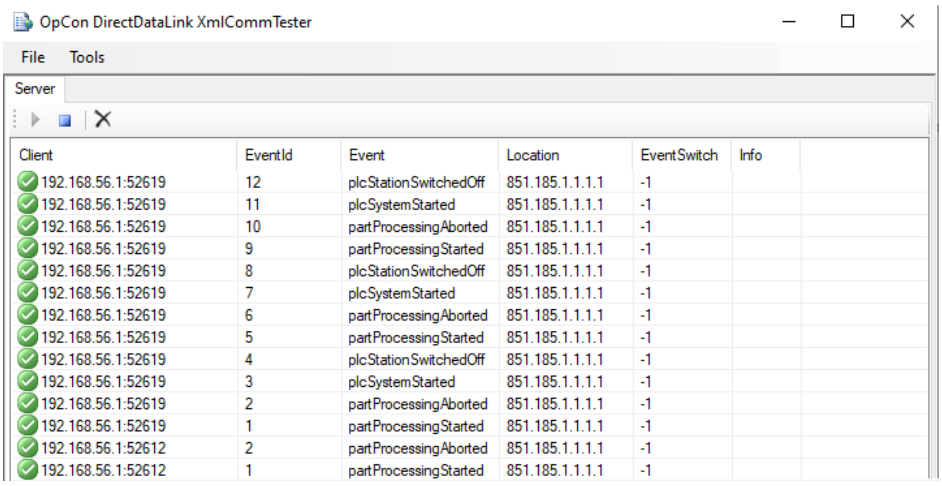
Apesar não ser possível a comprovação direta do envio dos telegramas para o MES, o seu envio foi comprovado pelo *Comm Tester*, figura 6.1. Já o conteúdo e a validade dos telegramas foi verificada e assegurada pelo engenheiro Duarte Almeida que é o responsável pelo *Nexeed MES* na *BOSCH Termotecnologia*. Para além de receber e armazenar

telegramas, este programa apresenta uma visualização básica sobre os telegramas recebidos, apresentando para cada telegrama o endereço IP e a porta de onde foi enviado, *eventid*, *Event*, *location*, *EventSwitch* e *info*.

O teste do funcionamento final em produtivo das aplicações decorreu durante uma manhã. Inicialmente o operador desligou a máquina, seguidamente as aplicações foram iniciadas. A partir do momento em que o computador se encontrava fisicamente ligado ao HMI, o operador executou o processo de iniciação da maquinagem. Após o equipamento ligar respondeu a 2 *pings* o que fez com que a aplicação *opcda_ bosch* iniciasse a monitorização e enviasse uma mensagem para aplicação *opcae_ bosch* para que esta também iniciasse a sua monitorização. Nesta etapa foi enviado o telegrama *plcSystemStarted*, um telegrama *plcError* com a informação de que a paragem de emergência se encontra ativa e um telegrama *plcModeChange* com o modo igual a MDI.

Seguidamente o operador selecionou o programa da peça a maquinar e colocou-o em funcionamento, o que levou ao envio de um telegrama *plcModeChange* com o modo igual a auto e um telegrama *plcError* devido à desativação da paragem de emergência. Durante o funcionamento do ciclos de maquinação foram enviados telegramas *plcToolChangeStarted* e *plcToolChanged* sempre que o equipamento iniciava e terminava, respetivamente, a mudança de ferramenta.

Na primeira rotação da mesa foi comprovado que não existiu envio de telegramas *PartProcessed*, pois trata-se da primeira rotação após o programa ter o status de *running*. Esta rotação serve para as primeiras peças a maquinar ficarem na secção de maquinação. Nesta primeira rotação apenas foram enviados telegramas *PartProcessingStarted*, sendo que existiu uma abertura correta do ficheiro *client.config* e uma associação do nome do programa em execução ao identificador das peças e ao número de peças por mesa. Durante a maquinagem de uma das mesas, o operador pressionou o botão de paragem de emergência que fez com que o estado do programa passasse a *interrupted* e que fossem enviados 4 telegramas *PartProcessingPaused*, pois eram maquinadas 4 peças por mesa. Noutra ocasião o operador cancelou o programa no HMI o que originou no envio de 4 telegramas *PartProcessingAborted*. Por fim o operador desligou o equipamento culminando no envio de um telegrama *PlcStationSwitchedOff*



Client	EventId	Event	Location	EventSwitch	Info
192.168.56.1:52619	12	plcStationSwitchedOff	851.185.1.1.1.1	-1	
192.168.56.1:52619	11	plcSystemStarted	851.185.1.1.1.1	-1	
192.168.56.1:52619	10	partProcessingAborted	851.185.1.1.1.1	-1	
192.168.56.1:52619	9	partProcessingStarted	851.185.1.1.1.1	-1	
192.168.56.1:52619	8	plcStationSwitchedOff	851.185.1.1.1.1	-1	
192.168.56.1:52619	7	plcSystemStarted	851.185.1.1.1.1	-1	
192.168.56.1:52619	6	partProcessingAborted	851.185.1.1.1.1	-1	
192.168.56.1:52619	5	partProcessingStarted	851.185.1.1.1.1	-1	
192.168.56.1:52619	4	plcStationSwitchedOff	851.185.1.1.1.1	-1	
192.168.56.1:52619	3	plcSystemStarted	851.185.1.1.1.1	-1	
192.168.56.1:52619	2	partProcessingAborted	851.185.1.1.1.1	-1	
192.168.56.1:52619	1	partProcessingStarted	851.185.1.1.1.1	-1	
192.168.56.1:52612	2	partProcessingAborted	851.185.1.1.1.1	-1	
192.168.56.1:52612	1	partProcessingStarted	851.185.1.1.1.1	-1	

Figura 6.1: Exemplo de teste de receção dos telegramas XML

6.1 Testes de Desempenho

Como forma de avaliar a subscrição de dados do servidor OPC *DA*, a aplicação *opcda_bosch* foi alvo de alterações sucessivas. Na base destas alterações está a identificação do limite máximo de itens que suportava subscrever. Este teste iniciou-se com 10 variáveis com incrementos de 5 variáveis em cada iteração. Os resultados destes testes demonstram que a aplicação funciona na perfeição na subscrição de um máximo de 40 variáveis. A partir deste valor o seu tempo de resposta torna-se muito elevado, cerca de 10 segundos. Quando o número de itens subscritos ultrapassa os 60, a aplicação não consegue correr e aborta. É importante referir que o funcionamento do HMI ocorre de maneira normal quando a aplicação aborta. A função *callback* que recebe as informações relativas à alteração do valor dos itens é assíncrona. Sempre que um item tem o seu valor alterado esta função é chamada e é executada assincronamente. No caso de muitos itens terem o seu valor alterado e estas alterações ocorrerem muito frequentemente e, ao mesmo tempo serão executadas várias funções assincronamente sempre no mesmo *thread*. Quando a quantidade de processamento necessária para a execução destas funções supera a capacidade de processamento de um *thread* do equipamento onde são executadas elas abortam ou deixam de responder. Uma alternativa seria utilizar diferentes *threads* para o processamento das *callbascks*.

Tal estratégia não se justifica aplicar nas aplicações deste projeto pois subscrevem poucos itens e a implementação de *multi-threading* torna-se complexa.

Envio de telegramas

Para analisar a capacidade de envio de telegramas por parte das aplicações foi mais uma vez alterado o código de ambas, de modo a implementar o envio de um número fixo de telegramas com o intuito de testar o tempo de envio de telegramas por parte das aplicações. A medição deste tempo teve por base a diferença entre os *TimeStamps* do primeiro e último programa, sendo que o objetivo deste teste era unicamente testar o *delay* entre o envio de telegramas das aplicações. Devido à impossibilidade de envio de telegramas para o *Nexeed* MES, o tempo de envio de telegramas entre a aplicação e o servidor não foi testado.

Para a realização deste teste foi utilizado o *Comm Tester*. Para a receção dos telegramas, ambas as aplicações e o *Comm Tester* foram executados no mesmo computador. Ambas as aplicações obtiverem tempos muito similares pelo que os resultados apresentados na figura 6.2 advêm de uma média de 10 testes, 5 para cada aplicação.

O tempo de transmissão dos telegramas desde a aplicação até ao servidor MES não tem influência nos dados, pois o servidor *Nexeed* utiliza o valor do item *TimeStamp* para preencher as suas bases de dados. Da análise da figura 6.2, conclui-se que o tempo médio para o envio de um telegrama é de 30 milissegundos, testado até um envio de 100 telegramas seguidos. Este valor diz respeito à diferença temporal entre o *TimeStamp* do primeiro e último telegrama. É um valor muito baixo e que se adequa perfeitamente ao trabalho.

O número máximo de telegramas seguidos que o sistema desenvolvido poderá ser obrigado a enviar são 4, e ocorre para os telegramas decorrentes da maquinação das peças, pois cada mesa pode possuir até 4 peças que iniciam e terminam a maquinação em simultâneo.

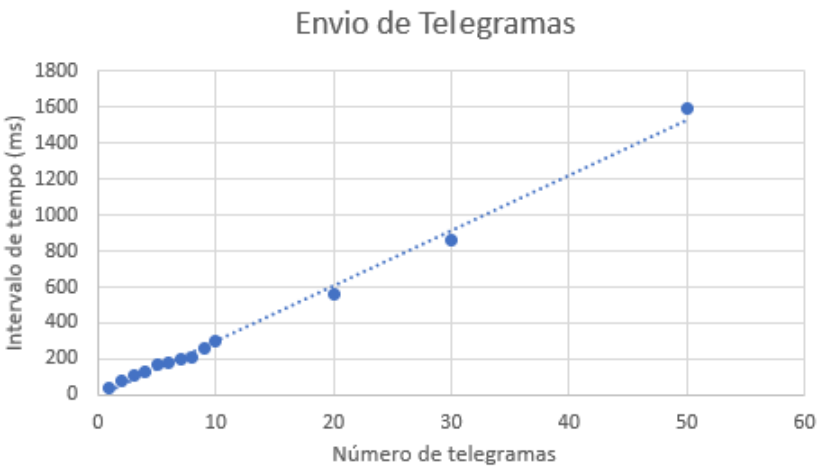


Figura 6.2: Gráfico do intervalo de tempo necessário para o envio de vários telegramas

Falha na comunicação entre aplicações

Como forma de teste de comunicação entre aplicações foi, testado o envio de mensagem de uma aplicação para outra, sendo que a aplicação recetora não estava a ser executada. Este teste validou os mecanismos de aviso à falta de resposta das mensagem enviadas por *IPC Pipes*, na forma de um aviso em *Text Box*.

Envio de telegramas

Durante os testes de funcionalidade de ambas as aplicações em 100% dos casos os telegramas foram enviados de forma correta e atempadamente e em nenhum dos casos existiu falhas no seu envio.

Após a desconexão do equipamento da energia elétrica e consequente conexão, ambas as aplicações demonstraram a capacidade de se desconectarem dos servidores OPC e efetuarem uma nova conexão. Este seria sem dúvida o fator mais crítico para ambas as aplicações, mas foi superado.

Já no que diz respeito a todos os outros telegramas, no decorrer dos testes funcionais foi comprovado o seu envio imediatamente após o item que lhes dava origem alterava o seu valor.

Capítulo 7

Conclusões

No presente de projeto foi desenvolvido um sistema que suporta a supervisão de um centro de maquinagem equipado com um comando numérico *Siemens* e que é capaz de comunicar os dados supervisionados ao sistema MES implementado na empresa *Bosch Termotecnologia* localizada em Aveiro.

O primeiro capítulo apresenta o problema e os objetivos propostos, a necessidade de tornar mais transparente todo o processo de maquinagem.

Para dar resposta ao problema foi efetuada uma grande pesquisa acerca de sistemas de monitorização de equipamentos CNC, sobre a arquitetura de comunicação presente no equipamento a monitorizar, e por fim, uma pesquisa com o objetivo de uma melhor compreensão de sistemas MES. Desta pesquisa surgiram várias arquiteturas passíveis de implementação, das quais foi selecionada a mais rápida e fiável, que passou pela utilização de duas aplicações *OPC clients*.

Pode-se concluir que o sistema desenvolvido dá resposta a todos os objetivos definidos neste projeto, sendo que a sua utilização permite uma maior transparência para todo o processo de maquinagem executado pelo equipamento em estudo, e que os dados disponibilizados quando integrados no MES permitem um aumento da eficiência do processo, possibilitando a implementação de manutenções preditiva, com recurso a tecnologias de inteligência artificial, identificação de tempos de paragens, desgastes de ferramentas, *Bottlenecks*, cálculos de OEEs, KPIs e comunicação ao operador dos dados de produção via *Andon*. Com este conjunto de dados recolhidos em tempo real torna-se possível alcançar uma diminuição do desperdício, o aumento da eficiência do equipamento, uma redução dos custos de produção e uma maior digitalização do processo que correspondem aos grandes desafios de qualquer empresa do século XXI.

Trabalhos Futuros Concluído o projeto constata-se que, apesar de o sistema desenvolvido dar resposta aos objetivos propostos, abre o caminho para muito trabalho a desenvolver.

Como referido anteriormente as aplicações desenvolvidas enviaram os telegramas para um sistema de simulação do *Nexeed MES* que foi validado pelo engenheiro Duarte Almeida, que é o coordenador do MES na empresa. No entanto, fica como ponto de trabalho futuro a ligação ao sistema de produção que por regras da empresa (nomeadamente gestão da rede de produção e gestão de alterações ao MES produtivo), não foi possível executar durante o período do projeto.

O próximo passo para a implementação do sistema será a alteração das aplicações para serem executadas como serviços *Windows* em servidores. Esta alteração permitiria uma maior simplicidade na execução de ambas as aplicações, que possuem um layout muito simplificado de modo a facilitarem a sua conversão em serviços.

A validação da ligação por OPC *DA* para recolha de dados durante o funcionamento de equipamentos CNC controlados por *Siemens 840D* com HMI *PCU 50.2*, abre o caminho para a identificação e recolha de muitas outras variáveis para além das utilizadas neste projeto. Com uma maior recolha de dados será possível controlar outros elementos do equipamento como consumos energéticos, maior precisão no desgaste de ferramentas e tempo de vida útil do liquido refrigerante.

Por fim, fica a faltar ainda um grande trabalho de tratamento dos dados, por parte da empresa, com o objetivo de identificar melhorias ao processo produtivo que culminem numa diminuição dos tempos de paragem e num aumento da eficiência do processo de maquinagem neste equipamento.

Dificuldades A maior dificuldade na execução deste projeto foi, sem dúvida a recolha de dados da CNC. Tratando-se de um equipamento *legacy*, desenvolvido há vários anos numa época em que fatores como digitalização de processos e facilidade no acesso dos dados de funcionamento ainda não estavam presentes nas prioridades de fabricantes deste tipo de produtos. Todas as possibilidades de recolha de informação, diretamente do controlo numérico, encontram-se pobremente documentadas e requerem muita investigação. Toda a documentação que a empresa possui acerca do equipamento encontra-se direcionada para o utilizador do equipamento. Dada esta situação, a recolha de informação sobre o *sofaware* e *hardware* do equipamento tornou-se difícil, existindo sempre muitas incertezas quanto à validade das informações recolhidas. Após a seleção da utilização dos servidores OPCs do HMI para a recolha de dados, o processo de desenvolvimento da conexão ao mesmo foi iterativo e exaustivo, pois todas as ferramentas de cliente OPC disponíveis ora não funcionam com *Siemens*, ora são pagas, pelo que o desenvolvimento das aplicação constituintes da solução foi efetuado inicialmente sem grandes certezas quanto a sua funcionalidade final. Qualquer intervenção no sistema produtivo *Bosch* carece de processos burocráticos e de aprovação, aplicando-se a intervenções em equipamentos produtivos, alterações à rede informática ou ao sistema MES. Tais burocracias inviabilizaram a conexão do sistema de monitorização à rede de produção e ao MES produtivo. De qualquer forma as comunicação MES foram devidamente testadas em simulador de receção de dados do *Nexeed MES* e validadas pelo coordenador *Nexeed MES*.

Bibliografia

[1] Wikipedia contributors. *Dynamic Data Exchange — Wikipedia, The Free Encyclopedia*. URL: https://en.wikipedia.org/w/index.php?title=Dynamic_Data_Exchange&oldid=978138177 (acedido em 02/05/2020).

[2] Wikipedia contributors. *Object Linking and Embedding — Wikipedia, The Free Encyclopedia*. URL: https://en.wikipedia.org/w/index.php?title=Object_Linking_and_Embedding&oldid=941456782 (acedido em 11/03/2020).

[3] Wikipedia contributors. *DCE/RPC — Wikipedia, The Free Encyclopedia*. URL: https://en.wikipedia.org/w/index.php?title=DCE/RPC&oldid=926926716 (acedido em 02/05/2020).

[4] Wikipedia contributors. *Firewall (computing) — Wikipedia, The Free Encyclopedia*. URL: https://en.wikipedia.org/w/index.php?title=Object_Linking_and_Embedding&oldid=941456782 (acedido em 02/06/2020).

[5] LATUNOV, Ivan. *Inter-Process Communication in .NET Using Named Pipes*. 2004. URL: https://www.codeproject.com/Articles/7177/Inter-Process-Communication-in-NET-Using-Named-P-2 (acedido em 03/06/2020).

[6] Bosch GmbH. *Nexeed Manufacturing Execution System*. URL: https://www.bosch-connected-industry.com/en/connected-manufacturing/single-solutions/nexeed-manufacturing-execution-system/ (acedido em 24/04/2020).

[7] Modern Machine Shops. "2017 EXECUTIVE SUMMARY". Em: (2017), pp. 1–24. URL: http://www.connectedfactoryglobal.com/global/wp-content/uploads/2017/09/Modern-Machine-Shop-Benchmarking-2017.pdf.

[8] Bosch GmbH. *A Bosch no mundo*. URL: https://www.bosch.pt/a-nossa-empresa/o-grupo-bosch-no-mundo/ (acedido em 07/04/2020).

[9] Wikipedia contributors. *Robert Bosch GmbH — Wikipedia, The Free Encyclopedia*. URL: https://en.wikipedia.org/w/index.php?title=Robert_Bosch_GmbH&oldid=948124598 (acedido em 07/04/2020).

[10] Bosch GmbH. *Bosch em Portugal*. URL: https://www.bosch.pt/a-nossa-empresa/bosch-em-portugal/ (acedido em 07/04/2020).

[11] Bosch GmbH. *Bosch Termotecnologia*. URL: https://www.bosch.pt/a-nossa-empresa/bosch-em-portugal/aveiro/ (acedido em 07/04/2020).

[12] Chiron. *Chiron, Manual do Utilizador*. 2006.

[13] Chiron. *Through the course of time*. URL: https://chiron.de/en/company/about-chiron/history (acedido em 15/04/2020).

[14] OPC Foundation. *The Industrial Interoperability Standard.* URL: https://opcfoundation.org/ (acedido em 22/03/2020).

[15] OPC Foundation. *What is OPC?* URL: https://opcfoundation.org/about/what-is-opc/ (acedido em 11/03/2020).

[16] Forumautomation contributors. *Introduction to OPC and OPC Architecture.* URL: https://automationforum.in/t/introduction-to-opc-and-opc-architecture/2701.

[17] FONSECA, Marcos. "Comunicação OPC - Uma abordagem prática". Em: *Metalurgia e Materiais* 59.SUPPL.5 (2003), pp. 3–7. ISSN: 01040898. URL: http://alvarestech.com/temp/smar/www.delt.ufmg.br/seixas/PaginaSDA/Download/DownloadFiles/OPCMarcosFonseca.PDF.

[18] OPC Foundation. "OPC Data Access Custom Interface Standard". Em: (2004), pp. 1–190. URL: www.opcfoundation.org.

[19] Advosol. "Alarms and Events Custom Interface". Em: *Event (London)* (2002). URL: https://advosol.com/OpcSpecs/OPCAE1.10Specification.pdf.

[20] SILVA, Débora. *OPC A&E: Entenda porque você deveria saber mais a respeito.* 2018. URL: https://www.logiquesistemas.com.br/blog/opc-ae/ (acedido em 23/03/2020).

[21] UKEssays. *The History Of Cnc Machines.* URL: https://www.ukessays.com/essays/information-technology/examining-the-history-of-cnc-machines-information-technology-essay.php (acedido em 15/04/2020).

[22] Wikipedia contributors. *History of numerical control — Wikipedia, The Free Encyclopedia.* URL: https://en.wikipedia.org/w/index.php?title=History_of_numerical_control&direction=prev&oldid=951851110 (acedido em 15/04/2020).

[23] PANKAJ, Mishra. *What is CNC Machine – Main Parts, Working, Block Diagram.* 2017. URL: https://www.mechanicalbooster.com/2017/01/what-is-cnc-machine.html (acedido em 17/04/2020).

[24] SANTOS, Humberto. *HISTÓRIA DO MES – MANUFACTURING EXECUTION SYSTEM.* 2016. URL: https://flowtech.pt/pt/blog/historia-mes-manufacturing-execution-system/ (acedido em 11/04/2020).

[25] Wikipédia contributors. "Material Requirement Planning — Wikipédia, a enciclopédia livre". Em: (2019). URL: https://pt.wikipedia.org/w/index.php?title=Material_Requirement_Planning&oldid=55592448.

[26] ISA. "ISA-95 DRAFT STANDARD Enterprise-Control System Integration Part 1 : Models and Terminology". Em: (2008), pp. 1–165.

[27] Wikipedia contributors. "Manufacturing execution system — Wikipedia, The Free Encyclopedia". Em: (2020). URL: https://en.wikipedia.org/w/index.php?title=Manufacturing_execution_system&oldid=950880493 (acedido em 11/06/2020).

[28] SANTOS, Humberto. "A diferença entre o MES e o ERP". Em: (2016). URL: https://flowtech.pt/pt/blog/diferenca-mes-erp/ (acedido em 11/06/2020).

[29] HARJA Herman, PRAKOSA Tri, YUWANA Yatna, NURHADI Indra e JANUARTHA Adrian. "Development of real time machine tools component utilization data acquisition for developing dynamic model of maintenance scheduling". Em: *E3S Web of Conferences* 130 (2019), pp. 1–10. ISSN: 22671242. DOI: 10.1051/e3sconf/201913001015. URL: https://www.e3s-conferences.org/articles/e3sconf/abs/2019/56/e3sconf_ic-amme2018_01015/e3sconf_ic-amme2018_01015.html.

[30] ZHENG Bo, XU Junjie, LI Helin, XING Junwei, ZHAO Huadong e LIU Guoning. "Development of Remotely Monitoring and Control System for Siemens 840D sl NC Machine Tool Using Snap 7 Codes". Em: 86.Eame (2017), pp. 108–112. DOI: 10.2991/eame-17.2017.26. URL: https://webcache.googleusercontent.com/search?q=cache:CmGo56hUiQgJ:https://download.atlantis-press.com/article/25875769.pdf+{\&}cd=2{\&}hl=pt-PT{\&}ct=clnk{\&}gl=pt.

[31] NARDELLA, Davide. *Step7 Open Source Ethernet Communication Suite*. URL: http://snap7.sourceforge.net/ (acedido em 22/02/2020).

[32] Machine Metrics. *Machine Monitoring & OEE Software*. URL: https://www.machinemetrics.com/machine-monitoring (acedido em 12/02/2020).

[33] ASLAN, Deniz. "Integration of Virtual and On-line Machining Process Control and Monitoring using CNC Drive Measurements". Tese de doutoramento. 2011.

[34] Softing. *uagate-840D*. URL: https://data-intelligence.softing.com/products/iot-gateways/uagate-840d/ (acedido em 14/02/2020).

[35] *MtLink*. URL: https://www.fanucamerica.com/products/cnc/cnc-software-solutions/mtlink-i (acedido em 15/02/2020).

[36] LIU Dan, SHOUDE Tang, GUANGHUA Xu, AILING Luo e XIAODONG Wang. "Research on data acquisition and analysis of circular tests on numerical control machine tools based on NC system". Em: *Proceedings of the 2011 2nd International Conference on Digital Manufacturing and Automation, ICDMA 2011* (2011), pp. 952–955. DOI: 10.1109/ICDMA.2011.236. URL: https://ieeexplore.ieee.org/document/6052069.

[37] WANG Wei, ZHANG Xinyu, LI Yan e LI Yong. "Open CNC Machine Tool's State Data Acquisition and Application Based on OPC Specification". Em: *Procedia CIRP* 56 (2016), pp. 384–388. ISSN: 22128271. DOI: 10.1016/j.procir.2016.10.061. URL: http://dx.doi.org/10.1016/j.procir.2016.10.061.

[38] Siemens. *SINUMERIK 840D sl NC variable and interface signals*. 2013, pp. 1–428.

[39] Siemens. "System Software for S7-300 / 400 System and Standard Functions". Em: (2010), pp. 13–30.

[40] Bosch Gmbh. "It shopfloor solutions". Em: (2017).

[41] LOPES Maria. "Building an Industry 4.0 Platform: The implementation of OpCon MES at AvP". Em: July (2017).

[42] PA-ATMO. *OpCon Xml Spezifikation V2.2*.

[43] Wikipedia contributors. *XML — Wikipedia, The Free Encyclopedia*. 2020. URL: https://en.wikipedia.org/w/index.php?title=XML&oldid=945023687 (acedido em 09/04/2020).

[44] W3Resource. *XML declarations.* URL: https://www.w3resource.com/xml/declarations.php (acedido em 29/04/2020).

[45] Wikipédia. *Lista de portas dos protocolos TCP e UDP — Wikipédia, a enciclopédia livre.* 2021. URL: https://pt.wikipedia.org/w/index.php?title=Lista_de_portas_dos_protocolos_TCP_e_UDP&oldid=58200778 (acedido em 02/06/2020).

[46] Siemens. *Commissioning Manual SINUMERIK PCU Basesoftware (IM6)*, pp. 8,27–29.

[47] GAVIN, Brady. *How to Set a Minimum Password Length in Windows 10.* 2020. URL: https://www.howtogeek.com/509156/how-to-set-a-minimum-password-length-in-windows-10/ (acedido em 02/06/2020).

Apêndices

Apêndice A

Telegramas *Nexeed*

Os diferentes tipos telegramas que são utilizados pelo *Nexeed* MES, são apresentados neste anexo. O tipo de telegrama é definido pelo elemento *Event*.

A.1 Event

Tabela A.1: Sub-elementos do *event*.

Elemento	Descrição
<*plcChangeOverStarted*>	Evento acionado no início de um processo de mudança de sequência.
<*plcChangeOver*>	Evento acionado no fim de um processo de mudança do PLC.
<*plcOperationModeChanged*>	Evento acionado quando o modo de operação do PLC mudou.
<*plcSystemStarted*>	Evento acionado sempre que o PLC é ligado.
<*plcStationSwitchedOff*>	Evento acionado imediatamente antes do PLC ser desligado.
<*plcError*>	Evento acionado sempre que um erro ocorre no PLC contêm informações especificas sobre o erro.
<*plcPartsMissingStarted*>	Evento acionado quando faltam peças.
<*plcPartsMissing*>	Evento acionado sempre que a falta de peças é corrigida.
<*plcJamStarted*>	Evento acionado quando o equipamento encrava.
<*plcJam*>	Evento acionado quando o equipamento deixa de estar encravado.
<*plcOperatorRequiredStarted*>	Evento acionado sempre que existe uma chamada do operador.
<*plcOperatorRequired*>	Evento acionado quando o operador chamado se apresenta.
<*plcShiftChanged*>	Evento acionado quando se dá uma mudança de turno.
Contínua na próxima página	

Tabela A.1 – Continuação da página anterior

Elemento	Descrição
<*plcChargeChanged*>	Evento acionado quando se dá uma mudança de lotes.
<*plcMaterialChangeStarted*>	Evento enviado pelo processo de fabricação imediatamente após a mudança de material.
<*plcMaterialChanged*>	Evento enviado quando a mudança de material acontece.
<*plcToolChangedStarted*>	Evento enviado antes de se executar uma mudança de ferramenta.
<*plcToolChanged*>	Evento enviado quando se executar a mudança de ferramenta.
<*plcLogIn*>	Evento enviado quando um utilizador faz *log in*.
<*plcLogOff*>	Evento enviado quando um utilizador faz *log off*.
<*partReceived*>	Evento enviado quando uma peça é recebida na estação.
<*partProcessingStarted*>	Evento acionado quando se inicia o processamento de uma peça.
<*partProcessingPaused*>	Evento acionado quando se dá uma pausa no processamento de uma peça.
<*partProcessingAborted*>	Evento acionado quando o processamento de uma peça é abortado.
<*partProcessed*>	Evento acionado quando é concluído o processamento de uma peça.
<*partDisplaced*>	Evento acionado quando uma peça é mudada durante o processamento em linha.
<*dataDownloadRequired*>	Evento acionado quando o equipamento solicita dados ao servidor.
<*dataUploadRequired*>	Evento acionado quando o equipamento solicita o envio de dados para o servidor.
<*result*>	Telegrama enviado pelo MES com o resultado de um processamento.
<*trace*>	Telegrama enviado pelo MES com o resultado de múltiplos processamentos.

Apêndice B

PLC Program Bolcks

Neste anexo são apresentados os blocos constituintes de um programa PLC.

B.1 DBs

Tabela B.1: Visão geral dos *Data Blocks* [38].

Número	**Dados**
1	Reservado para *Siemens*
2-5	Mensagens PLC
6-8	Programa básico
9	Interface para os ciclos de compilação do NC
10	Interface central NC
11	Interface modo grupo
12	Sistema de transporte e ligação ao computador
13-14	Reservado para *Siemens*
15	Programa básico
16	Definição dos serviços PI (Process Integration)
17	Código da versão
18	Interface SPL (Safe Programmable Logic)
19	Interface MMC (Man Machine Communication)
20	Dados da máquina PLC
21-30	Interface dos canais NC
31-61	Reservado para a interface dos eixos e árvores
71-74	Gestão de ferramentas
75-76	Descodificação do grupo M
77	Buffer de gestão de ferramentas
78-80	Reservado para *Siemens*
1000	Controlo energético
1002-1070	Reservado para *Siemens*
1071	Armazém de ferramentas
1072	Árvore com várias ferramentas
1074-1099	Reservado para *Siemens*

B.2 FCs

Tabela B.2: Visão geral dos blocos de FC

Número	**Função**
0	Reservado para *Siemens*
2	Programa básico, ciclo peça
3	Programa básico, alarmes
5	Programa básico, diagnóstico de alarmes
7	Troca de ferramentas em armazém de ferramentas circular
8	Gestão de ferramentas
9	Programas assíncronos
10	Alarme e mensagens
12	Interface para o uso de funções auxiliares
13	Controlo de exibição para unidade portátil
15	Posicionamento dos eixos
16	Indexação dos eixos
18	Controlo da árvore
19	Distribuição dos sinais do MCP e do software operacional no interface
21	Troca de dados entre o PLC e o NC
22	Seleção da direção para a gestão de ferramentas
24	Transferência dos sinais do MCP para a interface
25	Distribuição do painel de controlo da máquina
30-999	Livre

B.3 FBs

Tabela B.3: Visão geral dos blocos FB.

Número	**Função**
0-29	Reservado para *Siemens*
1	Programa básico, arranque
2	Leitura das variáveis NC
3	Escrita das variáveis NC
4	Serviços PI
5	Leitura das variáveis GUD
7	Geral serviços PI
29	Diagnóstico para recolha de sinais e desencadeamento de dados
36-255	Livre

Apêndice C

Setup

Neste anexo são descritas as alterações efetuadas no HMI PCU50 e no computador onde estão alojadas as aplicações para a conexão das aplicações com os seus servidores OPCs.

C.1 HMI (PCU50)

Alterações no HMI. Para que as aplicações **opcae_boch** e **opcda_boch** possam aceder ao servidor *opc* via uma comunicação TCP/IP é necessário desativar ou modificar a Firewall. A desativação completa é desaconselhada, na medida em que desliga totalmente a proteção de rede. A opção mais segura, neste caso, é excluir da intervenção da Firewall os programas responsáveis pelo servidor OPC DA e pelo OPC AE, sendo também necessário excluir a porta 135, pois é a porta utilizada pelo serviço RPC [45].

Passos para a configuração da Firewall do HMI PCU50:

- **1:** Se ligado, desligar o equipamento.
- **2:** Ligar o PCU.
- **3:** Quando aparecer *"Please select operating system to start"* pressionar a a tecla seta para baixo até a linha debaixo de *Sinumerik* estar destacada a branco.
- **4:** Confirmar, pressionando a tecla *input*.
- **5:** No *Service Menu* selecionar a opção 4 "Start Windows Service Mode"[46].
- **6:** Selecionar a opção "*Standard Windows (without HMI)*"
- **7:** PCU executa um *Restart*.
- **8:** Utilizador "auduser"[46].
- **9:** Password "SUNRISE"[46].
- **10:** Esperar que se execute o carregamento completo do *Windows XP*.
- **11:** *Start -> Control Painel - >Windows Firewall*
- **12:** Menu *Exceptions*.

- **13:** *Add Port.. -> Port Number = 135.*
- **14:** *Add program -> path "F:\mmc2\opc\dataaccess\SOPC_MachineSwitch.exe"*
- **15:** *Add program -> path "F:\mmc2\opc\alarmevent\OPCSinumerikAlarm.exe"*
- **16:** Selecionar "OK"
- **17:** *Restart*

Os diretórios dos programas a excluir poderão ser diferentes consoante a versão do equipamento.

C.2 Servidor

Alterações a efetuar no computador/servidor onde estão alojadas as aplicações.

C.2.1 Desativação dos requisitos para passwords Windows 10

Passos para a desativação do tamanho mínimo e dos requerimentos de complexidade das palavras passe no *Windows 10* [47]:

- **1:** Pressionar as teclas *Win* e *R* simultaneamente.
- **2:** Digite *"gpedit.msc"* e pressione a tecla *Enter*.
- **3:** *Computer Settings -> Windows Settings -> Security Settings -> Account Policy -> Password Policy.*
- **4:** Desativar a política *"Password must meet complexity requirements"*.
- **5:** Definir como 0 o parâmetro *"Minimum password lenght"*.

C.2.2 Definição manual do endereço IP

Passos para a configuração de rede do servidor:

- **1:***Start -> Settings -> Network & Internet -> Change Adapter Settings.*
- **2:** *Right Click in Ethernet -> Properties -> Select Internet Protocol Version 4 (TCP/IP).*
- **3:** *Properties -> Click on Use the following IP address.*
- **4:** *Select Use the following IP address.*
- **5:** *Use an IP in same network as HMI. Example HMI = 192.168.1.16 server IP = 192.168.1.XX with XX between 0 and 255.*
- **6:** *Press Tab and the Subnet Mask section will populate with default numbers.*
- **7:** *Hit OK twice.*

Apêndice D

Ficheiros XML

Neste anexo são apresentados os ficheiros no formato XML usados no projeto.

D.1 PartNumber

Código Fonte D.1: Ficheiro partnumber.xml

```
<?xml version="1.0" encoding="utf-8"?>
<!--<This file is for configuration of the opcda_bosch>-->
<!--<Configurations acts as an database for the parts produced in the
    chiron FZ15>-->
<!--<Each entry associates the name of the program with the
    partidentifier and the number of parts produced per machine cycle(
    parts in table)>-->
<root>
        <configurations>
                <part partidentifier="8738710" programname="
                    _N_MANIFOLD_160KBTU_8738710_MPF" partspercycle="2"/>
                <part partidentifier="8738703" programname="
                    _N_MAN18GPL_8738703_MPF" partspercycle="2"/>
                <part partidentifier="8738723" programname="
                    _N_MAN18GN_8738723_MPF" partspercycle="2"/>
                <part partidentifier="8738718" programname="
                    _N_MAN15GPL_8738718_MPF" partspercycle="4"/>
                <part partidentifier="8738719" programname="
                    _N_MAN15GN_8738719_MPF" partspercycle="2"/>
                <part partidentifier="8738722" programname="
                    _N_MAN12GPL_8738722_MPF" partspercycle="4"/>
                <part partidentifier="8738725" programname="
                    _N_MAN12GN_8738725_MPF" partspercycle="4"/>
                <part partidentifier="8738713" programname="
                    _N_MANIFOLD_199KBTU_8738713_MPF" partspercycle="2"/>
        </configurations>
</root>
```

D.2 Telegramas

D.2.1 Plc System Started

Código Fonte D.2: Telegrama PlcSystemStarted

```
<?xml version="1.0" encoding="utf-8" standalone="yes"?>
<root>
        <header eventId="1" eventName="plcSystemStarted" version="1.0"
            eventSwitch="-1" timeStamp="2020-05-28T16:12:51.7537675+01:00"
            >
                <location lineNo="851" statNo="185" statIdx="1" fuNo="1"
                    workPos="1" toolPos="1" application="CNC" processName=
                    "ChironTeste" processNo="1000" />
        </header>
        <event>
                <plcSystemStarted />
        </event>
        <body />
</root>
```

D.2.2 Plc Station SwitchedOff

Código Fonte D.3: Telegrama plcStationSwitchedOff

```
<?xml version="1.0" encoding="utf-8" standalone="yes"?>
<root>
        <header eventId="5" eventName="plcStationSwitchedOff" version="
            1.0" eventSwitch="-1" timeStamp="2020-05-28T19
            :56:45.8660183+01:00">
                <location lineNo="851" statNo="185" statIdx="1" fuNo="1"
                    workPos="1" toolPos="1" application="CNC" processName=
                    "ChironTeste" processNo="1000" />
        </header>
        <event>
                <plcStationSwitchedOff />
        </event>
        <body />
</root>
```

D.2.3 Part Processing Started

Código Fonte D.4: Telegrama partProcessingStarted

```
<?xml version="1.0" encoding="utf-8" standalone="yes"?>
<root>
        <header eventId="1" eventName="partProcessingStarted" version="
            1.0" eventSwitch="-1" timeStamp="2020-05-28T19
            :56:45.7104361+01:00">
                <location lineNo="851" statNo="185" statIdx="1" fuNo="1"
                    workPos="1" toolPos="1" application="CNC" processName=
                    "ChironTeste" processNo="1000" />
        </header>
        <event>
                <partProcessingStarted identifier="8738703257" />
        </event>
        <body />
</root>
```

D.2.4 Part Processing Paused

Código Fonte D.5: Telegrama partProcessingPaused

```
<?xml version="1.0" encoding="utf-8" standalone="yes"?>
<root>
        <header eventId="5" eventName="partProcessingPaused" version="1.0
            " eventSwitch="-1" timeStamp="2020-05-28T19
            :53:52.1433647+01:00">
                <location lineNo="851" statNo="185" statIdx="1" fuNo="1"
                    workPos="1" toolPos="1" application="CNC" processName=
                    "ChironTeste" processNo="1000" />
        </header>
        <event>
                <partProcessingPaused identifier="8738703257" />
        </event>
        <body />
</root>
```

D.2.5 Part Processing Aborted

Código Fonte D.6: Telegrama partProcessingAborted

```
<?xml version="1.0" encoding="utf-8" standalone="yes"?>
<root>
        <header eventId="4" eventName="partProcessingAborted" version="
            1.0" eventSwitch="-1" timeStamp="2020-05-28T19
            :53:52.1074628+01:00">
                <location lineNo="851" statNo="185" statIdx="1" fuNo="1"
                    workPos="1" toolPos="1" application="CNC" processName=
                    "ChironTeste" processNo="1000" />
        </header>
        <event>
                <partProcessingAborted identifier="8738703257" />
        </event>
        <body />
</root>
```

D.2.6 Part Processed

Código Fonte D.7: Telegrama partProcessed

```
<?xml version="1.0" encoding="utf-8" standalone="yes"?>
<root>
        <header eventId="6" eventName="partProcessed" version="1.0"
            eventSwitch="-1" contentType="3" timeStamp="2020-05-28T19
            :53:52.1683362+01:00">
                <location lineNo="851" statNo="185" statIdx="1" fuNo="1"
                    workPos="1" toolPos="1" application="CNC" processName=
                    "ChironTeste" processNo="1000" />
        </header>
        <event>
                <partProcessed identifier="8738703257" />
        </event>
        <body>
                <structs>
                        <resHead result="1" typeNo="1111111111" typeVar="
                            0815" workingCode="0" nioBits="0" />
                </structs>
        </body>
</root>
```

D.2.7 Plc Tool Changed Started

Código Fonte D.8: Telegrama plcToolChangedStarted

```
<?xml version="1.0" encoding="utf-8" standalone="yes"?>
<root>
        <header eventId="5" eventName="plcToolChangeStarted" version="1.0
            " eventSwitch="-1" timeStamp="2020-05-28T16
            :13:37.8227025+01:00">
                <location lineNo="851" statNo="185" statIdx="1" fuNo="1"
                    workPos="1" toolPos="1" application="CNC" processName=
                    "ChironTeste" processNo="1000" />
        </header>
        <event>
                <plcToolChangeStarted identifier="29" />
        </event>
        <body />
</root>
```

D.2.8 Plc Tool Changed

Código Fonte D.9: Telegrama plcToolChanged

```
<?xml version="1.0" encoding="utf-8" standalone="yes"?>
<root>
        <header eventId="6" eventName="plcToolChanged" version="1.0"
            eventSwitch="-1" timeStamp="2020-05-28T16:13:38.6466159+01:00"
            >
                <location lineNo="851" statNo="185" statIdx="1" fuNo="1"
                    workPos="1" toolPos="1" application="CNC" processName=
                    "ChironTeste" processNo="1000" />
        </header>
        <event>
                <plcToolChanged identifier="29" />
        </event>
        <body />
</root>
```

D.2.9 Plc Error

Código Fonte D.10: Telegrama plcError

```
<?xml version="1.0" encoding="utf-8" standalone="yes"?>
<root>
        <header eventId="1" eventName="plcError" version="1.0"
            eventSwitch="-1" contentType="3" timeStamp="2020-05-28T16
            :12:53.6597865+01:00">
                <location lineNo="851" statNo="185" statIdx="1" fuNo="1"
                    workPos="1" toolPos="1" application="CNC" processName=
                    "ChironTeste" processNo="1000" />
        </header>
        <event>
                <plcError errorNo="700054" errorText="PREPARAR␣MANUTENCAO
                    " errorType="1" errorState="1" modeOn="true"
                    operationMode="1" typeNo="" typeVar="" typeVersion=""
                    chainNo="1" />
        </event>
        <body>
                <items>
                        <item name="Cookie" value="8999552" dataType="8"
                            />
                        <item name="EventCategory" value="Condition"
                            dataType="8" />
                        <item name="Severity" value="192" dataType="8" />
                </items>
                <structs />
        </body>
</root>
```

D.2.10 Plc Operation Mode Changed

Código Fonte D.11: Telegrama plcOperationModeChanged

```
<?xml version="1.0" encoding="utf-8" standalone="yes"?>
<root>
        <header eventId="6" eventName="plcOperationModeChanged" version="
            1.0" eventSwitch="-1" contentType="3" timeStamp="2020-05-28T20
            :04:59.8732501+01:00">
                <location lineNo="851" statNo="185" statIdx="1" fuNo="1"
                    workPos="1" toolPos="1" application="CNC" processName=
                    "ChironTeste" processNo="1000" />
        </header>
        <event>
                <plcOperationModeChanged modeOn="true" operationMode="2"
                    typeNo="" typeVar="" typeVersion="" shift="0" charge="
                    " specPrgNo="0" />
        </event>
        <body>
                <items>
                        <item name="Mode␣Description" value="Auto"
                            dataType="8" />
                </items>
                <structs />
        </body>
</root>
```