Università degli Studi di Ferrara

DOTTORATO DI RICERCA IN
SCIENZE DELL'INGEGNERIA
INGEGNERIA DELL'INFORMAZIONE

CICLO XXVII

COORDINATORE Prof. Stefano Trillo

e-Maintenance of Smart Connected Machines in the Industrial Internet of Things

Settore Scientifico Disciplinare ING-INF/05

Dottorando
Dott. Roberto Lazzarini

Tutore
Prof. Cesare Stefanelli

Anni 2012/2014

Keywords

ICT, e-Maintenance, Internet, IoT, Machines

Contents

List of Acronyms and Abbreviations

6LoWPAN	IPv6 over Low power Wireless Personal Area Networks
AI	Artificial Intelligence
AIS	Artifical Immune Systems
AMQP	Advanced Message Queuing Protocol
API	Application Programming Interface
ARIB	Association of Radio Industries and Businesses
ATIS	Alliance for Telecommunications Industry Solutions
CAD	Computer-Aided Design
CAM	Computer Aided Manufacturing
CASIP	Computer Aided Safety and Industrial Productivity
CBM	Condition-Based Maintenance
CCSA	China Communications Standards Association
CDS	Configuration Data Set
CMMS	Computerized Maintenance Management System
CoAP	Constrained Application Protocol
CORBA	Common Object Request Broker Architecture
COTS	Commercial Off-The-Shelf
CPU	Central Processing Unit
CRM	Customer Relationship Management
D2B	Device-to-Business
DBMS	DataBase Management System
DCM	Data Collection and Manipulation
DD	Data-Driven
DDS	Data Distribution Service
DoD	Department of Defense
DPM	Diagnostics and Prognostics Module
DPSA	Distributed Prognosis System Architecture
DSL	Domain-Specific Language
DTLS	Datagram Transport Layer Security
DYNAMITE	Dynamic Decisions in Maintenance
EIS	Engineering Immune Systems
ETSI	European Telecommunications Standards Institute

EU	European Union
FMECA	Failure Mode Effects and Criticality Analysis
GPLR	Government Purpose License Rights
GPRS	General Packet Radio Service
GPS	Global Positioning System
GSM	Global System for Mobile communications
HAZOP	HAZard OPerability
HGI	Home Gateway Initiative
HTTP	Hypertext Transfer Protocol
HW/SW	Hardware/Software
ICAS	Integrated Condition Assessment System
ICT	Information and Communication Technologies
IEC	International Electrotechnical Commission
IEEE	Institute of Electrical and Electronic Engineers
IERC	European Research Cluster on the Internet of Things
IETF	Internet Engineering Task Force
IIC	Industrial Internet Consortium
IIoT	Industrial Internet of Things
IoE	Internet of Everything
IoT	Internet of Things
IoT-GSI	Global Standards Initiative on Internet of Things
IMS	Intelligent Maintenance Systems
IP	Internet Protocol
IPSO	Internet Protocol for Smart Objects
IPv6	Internet Protocol version 6
ISO	International Organization for Standardization
ITU	International Telecommunication Union
ITU-T	ITU Telecommunication Standardisation Sector
KASEM	Knowledge and Advanced Services for e-Maintenance
LAN	Local Area Network
LRU	Line Replaceable Unit
LTE	Long Term Evolution
LWM2M	Lightweight M2M
M2M	Machine-to-Machine
MC	Monitoring Controller

MDSS	Maintenance Decision Support System
MEMS	MicroElectroMechanical Systems
MES	Manufacturing Execution System
MIMOSA	Machinery Information Management Open System Alliance
MMI	Man–Machine Interface
MPROS	Machinery Prognostics and Diagnostics System
MQTT	Message Queue Telemetry Transport
NASA	National Aeronautics and Space Administration
OASIS	Organization for the Advancement of Structured Information Standards
OEM	Original Equipment Manufacturer
OIC	Open Interconnect Consortium
OMA	Open Mobile Alliance
OMG	Object Management Group
OSI	Open Systems Interconnection
PaaS	Platform-as-a-Service
PC	Personal Computer
PDA	Personal Digital Assistant
PHM	Prognostics and Health Management
PI	Presentation Interface
PLC	Programmable Logic Controller
PLM	Product Lifecycle Management
PSS	Product-Service System
RCM	Reliability-Centered Maintenance
RDS	Remote Diagnostics System
ReST	Representational State Transfer
RM	Reporting Module
R&D	Research and Development
RUL	Remaining Useful Life
SaaS	Sotfware-as-a-Service
SBC	Single-Board Computer
SCADA	Supervisory Control And Data Acquisition
SDK	Software Development Kit
SDO	Standards Developing Organization
SEMATECH	Semiconductor Manufacturing Technology
SMMART	System for Mobile Maintenance Accessible in Real Time

SOA	Service-Oriented Architecture
SOAP	Simple Object Access Protocol
SSL	Secure Sockets Layer
TCP	Transmission Control Protocol
TIA	Telecommunications Industry Association
TLS	Transport Layer Security
TPM	Total Productive Maintenance
TTA	Telecommunications Technology Association
TTC	Telecommunication Technology Committee
UDP	User Datagram Protocol
UGW	Universal GateWay
UMTS	Universal Mobile Telecommunications System
W3C	World Wide Web Consortium
WAN	Wide Area Network
WSDF	Web-Services-based e-Diagnostics Framework
WSN	Wireless Sensor Network
XML	eXtensible Markup Language

Abstract

e-Maintenance expresses the emerging synthesis of two considerable trends in today's society: the rapid development of information and communication technology (ICT), and the growing importance of maintenance as a key strategy for managing the product life cycle.

In this research I studied and analyzed novel ICT models and technologies that can support innovative methods of e-Maintenance for the management of machines in the Industrial Internet of Things (IIoT). In particular, I designed a platform systems for the automatic and remote management of processes and/or machines, distributed throughout a region and the development of a specific operating platform in the IIoT.

I expect my efforts made it possible to identify the technologies that enable intelligent factories, as well as new business and manufacturing models. As of today, significant results include publications and patents. The research project has also driven the design of Carpigiani operating platform able to confirm how e-Maintenance has reach a level of confidence worthy of being successfully applied to machines, processes, and industrial systems, guaranteeing rapid, cost-efficient, and reliable implementation.

In this dissertation I initially analyze the context of the Industrial Internet of Things and the critical aspects that characterize its development. Then, I describe the subject of e-Maintenance and present the Carpigiani project. Finally, I studied the main related works, and draw conclusions.

Sommario

e-Maintenance esprime proprio il concetto emergente di sintesi di due rilevanti tendenze nella società di oggi ovvero il rapido sviluppo delle tecnologie dell'informazione e della comunicazione (ICT) e la crescente importanza della manutenzione come strategia chiave nella gestione del ciclo di vita dei prodotti.

In questo contesto si inserisce il presente progetto di ricerca, i cui obiettivi sono stati lo studio e l'analisi di modelli e tecnologie ICT finalizzati a supportare nuovi metodi di e-Maintenance, relativi a sistemi per la gestione automatica e il controllo remoto di processi e/o macchine, distribuiti su area geografica e lo sviluppo di una specifica piattaforma operativa nella Industrial Internet of Things (IIoT).

Il lavoro svolto nell'ambito della presente tesi ha consentito di individuare le tecnologie abilitanti la fabbrica intelligente e nuovi modelli di business e manufacturing.

I risultati conseguiti sono stati significativi sia per quanto riguarda pubblicazioni che deposito di brevetti. Il presente progetto di ricerca si è inoltre tradotto in una piattaforma operativa Carpigiani in grado di confermare come la e-Maintenance abbia realmente raggiunto un livello di confidenza tale da poter essere applicata con successo a macchine, processi e sistemi industriali garantendo implementazioni rapide, costo-efficienti e affidabili.

Nel presente lavoro si è analizzato inizialmente il contesto della Industrial Internet of Things e le principali criticità che ne caratterizzano lo sviluppo. Successivamente si è indagato il tema della e-Maintenance ed è stato descritto il progetto Carpigiani. Infine sono stati studiati i riferimenti in letteratura e le principali applicazioni esecutive.

Introduction

This project deals with various open questions regarding the interactions between machines and between humans and machines. A new generation of networked, intelligent products is radically changing not only the technical and technological context – the hardware aspect – but also the ways in which they communicate and cooperate. This involves a rethinking of the management of the full product life cycle (construction, use, storage, end of life or reuse), of which maintenance is an essential phase. Maintenance is tightly tied to strategic aspects like competitiveness and profitability, value retention, safety, sustainability and environment, to mention just a few of the most important. Maintenance and sustainable development, in particular, share a platform of values and objectives destined to assume an important position in future industrial scenarios and to guide associated strategic choices.

On a parallel track, today, thanks to tools made available through the internet and new computer technologies, it seems possible to realize the Internet of Things (IoT), eliminating barriers between computers and the real world, combining information and matter into a single entity. The rise of the Internet of Things in the industrial sector (relating to objects that mainly impact production processes and systems) represents one of the main drivers of future innovation and, already today, is redesigning company structures and revolutionizing the competitive landscape. Research presented by Accenture at the recent World Economic Forum Annual Meeting 2015 quantifies the market for Industrial Internet of Things (IIoT) at 14.2 trillion dollars by 2030. Nonetheless, the same report illustrates how inadequate infrastructures, competence, and institutions could frustrate these opportunities.

e-Maintenance expresses the emerging synthesis of these two considerable trends in today's society, the rapid development of information and communication technology (ICT), and the growing importance of maintenance as a key strategy for managing the product life cycle.

This research project reflects this new reality, its objectives being the study and analysis of ICT models and technologies that can support to new methods of e-Maintenance, systems for the automatic and remote management of processes and/or machines, distributed throughout a region and the development of a specific operating platform.

More precisely, the research had the following specific objectives:

- Understand how ICT models and technologies can drive and condition the design and implementation of e-Maintenance methodologies.

- Examine e-Maintenance trends, scenarios, industrial requirements, critical applications, technological solutions, standards, and protocols relating to future needs of distributed maintenance.

- Identify methods, architecture, standards, protocols, and ICT tools able to support applications and development of e-Maintenance solutions, and identify those which are potentially more promising (due to the significance of their benefits, maturity of the technology, growth potential, and so on).

- Analyze the value created by new models and ICT technologies in the most advanced e-Maintenance application environments.

- Identify case studies (feasibility studies, field tests, pilot projects, applications) with particular focus on ICT and structure, and evaluate application scenarios.

- Develop and implement an e-Maintenance platform on the Industrial Internet of Things.

The efforts made it possible to identify the technologies that enable intelligent factories, as well as new business and manufacturing models.

As of today, in terms of impact, the work presented in this thesis produced conference and journal publications, a contribution in volume and patent applications. The research project has also translated into an operating platform able to confirm how e-Maintenance has reach a level of confidence worthy of being successfully applied to machines, processes, and industrial systems, guaranteeing rapid, cost-efficient, and reliable implementation. Moreover this work has greatly influenced the design and implementation of the new after-sales service of Carpigiani Group, which the company considers one of its primary competitive differentiators.

In this dissertation I initially analyzed the context of the Industrial Internet of Things and the critical aspects that characterize its development (Chapter 1). Then, I introduce the concept of e-Maintenance and describe how modern ICT solutions are enabling its adoption for the management of large scale installations of low-cost automated machines (Chapter 2). Hence, I then present the Teorema Project, the Carpigiani integrated solution for e-Maintenance (Chapter 3). Finally, I analyzed the main related works (Chapter 4), and draw conclusions (Chapter 5).

Chapter 1

The Industrial Internet of Things

1.1. Introduction and Definition

The Internet of Things (IoT) is a concept that refers to the interconnection of uniquely identifiable (physical and virtual) objects within the existing Internet infrastructure. IoT offers advanced connectivity of devices, systems, and services that goes beyond machine-to-machine communications (M2M) and covers a variety of protocols, domains, and applications (Holler et al., 2014).

Things, in the IoT, are arbitrary objects existing in the world and refer to a wide variety of devices, sensors, and machines such as household appliances, biosensors, and low-cost automated machines. The term Internet of Things (IoT) was first introduced by Procter & Gamble employee Kevin Ashton in 1999 in the context of logistics and supply chain management (Ashton, 2009). Recently, however, the definition has been more inclusive covering wide range of applications and IoT is moving from futuristic vision to tangible reality.

The European Research Cluster on the Internet of Things (IERC) defines the IoT as a dynamic global network infrastructure with self-configuring capabilities based on standard and interoperable communication protocols where physical and virtual "things" have identities, physical attributes and virtual personalities, use intelligent interfaces and are seamlessly integrated into the information network (Vermesan and Friess, 2014).

The Industrial Internet of Things (IIoT) is a significant subset of the IoT and is connecting intelligent industrial *Things* (physical sensors, devices and machines) with the *Internet* and, by applying Big Data analytics, is turning information into powerful new insight and intelligence. Requirements for the Industrial IoT are a subset of the main IoT requirements. In fact, the IIoT needs simple, inexpensive and easy-to-install nodes but the consequences for communications failures can be much worse than in the case of the traditional Internet.

The recent developments in ICT will be fundamental for the coordination and synchronization between information and materials in the Industrial Internet of Things. Major ICT market players are developing concepts for industrial solutions using evocative terms such as Cisco *Internet of Everything* (http://www.cisco.com), GE *Industrial Internet* (http://www.ge.com), IBM *Smarter Planet* (http://www.ibm.com/smarterplanet), Echelon *IzoT* (http://www.echelon.com/izot-platform).

The Industrial Internet of Things introduces new technical challenges, e.g., the IIoT architecture requires a much more organic approach compared with standard networking, constrained devices can't bear traditional network protocols and, generally speaking, the system need to build up reliable information from a large number of potentially individually unreliable sources. The issue of the standards and protocols applicable is therefore central to the development of the new technology.

1.2. Standardization Support for the Industrial Internet of Things

Standards are needed for interoperability, manageability, security and effective performances between heterogeneous technical solutions on a global scale, thereby eliminating fragmentation and lock-in syndromes, as well as enabling new services and business models. Standardization is a critical success factor for market development, but it may threaten innovation and inhibit change when standards are accepted by the market. Therefore, a real and effective standardization implies that requirements from the various industries using and deploying IIoT-based solutions are considered, both in their specific field as well as for cross sector dependencies and mutual relations. Most current standardization activities are related to fragmented architectures and there are no coherent unifying concepts; others initiatives are confined to very specific verticals representing islands of disjointed or redundant development. Unlike in the Internet ecosystem, currently there are no agreements on a universal set of technical standards for the IoT.

Probably, the main reason is that IoT projects have to deal with physical properties depending on concrete applications and resulting in a large amount of alternative technological solutions. A second reason may be related to the fact that most of the manufacturing solutions are base on proprietary technology.

The prominent groups of companies are developing competing technology standards to support IoT. The relevant industry consortiums and organizations are described below.

a. AllSeen Alliance

Founded: 2013

Website: http://www.allseenalliance.org

Members: 112 (Electrolux, Haier, LG, Microsoft, Panasonic, Qualcomm, Sharp, Silicon Image, Sony, Technicolor, TP-Link, AT&T, Bosch, Cisco, D-Link, HTC, Symantec, etc.)

Allseen Alliance was announce by Qualcomm in 2013 as an open source foundation for their Alljoyn framework, a solution to allow virtually every device to talk to each other, including many that have no connectivity or intelligence, and share resources and information across networks, and operating system. AllJoyn enables ad hoc, proximity based, peer-to-peer, bearer agnostic networking between devices and applications.

Qualcomm has led the development of this project since 2011 then in 2013 they decided Alljoyn need a better governance model and so they created a foundation.

The AllSeen Alliance is a nonprofit consortium dedicated to enabling and driving the widespread adoption of products, systems and services that support the Internet of Everything (IoE) with an open, universal development framework and the Alljoyn framework is an open source project managed by the Linux Foundation. The AllSeen Alliance recognizes that no one company can accomplish the level of interoperability required to support the IoE and therefore depends on its members to contribute software and engineering resources as part of their collaboration to advance its development.

b. Continua Health Alliance (Continua)

Founded: 2006

Website: http://www.continuaalliance.org/about-continua

Members: 180 (Intel, IBM, Fujitsu, A&D Medical, Institute for Information Industry, Novartis, Omron, Oracle, Orange, Panasonic, Philips, Qualcomm, Roche, JDRF, Samsung, Sharp, TATRC, UnitedHealth Group, etc.)

Continua is a nonprofit open industry consortium of healthcare and technology companies and was a founding member of Personal Connected Health Alliance which was launched in February 2014 with other founding members mHealth SUMMIT and HIMSS. Continua is focused on interoperability of healthcare and medical devices and cooperates with major standards developing organizations. In December 2013 the Continua Design Guidelines for e-health interoperability were adopted by ITU-T.

c. Eclipse IoT

Founded: 2013

Website: http://iot.eclipse.org

Members: 22 (Eurotech, IBH Systems, Sierra Wireless, etc.)

The Eclipse IoT community includes 17 different open source projects (e.g., Mosquitto, Paho, Kura, Mihini, Eclipse SCADA, etc.) and 21 organizations participating in the IoT Working Group. The goal of Eclipse IoT is to provide open IoT frameworks and open source implementations of IoT standards. The IoT Working Group is supporting lightweight two-way communications between devices and server and propose open source implementations for protocols already adopted in M2M fields such as MQTT and CoAP.

d. Home Gateway Initiative (HGI)

Founded: 2004

Website: http://www.homegatewayinitiative.org

Members: 47 (Deutsche Telekom, BT, KPN, NTT, Orange, TeliaSonera, Telecom, etc.)

The HGI, founded by major broadband service providers and joined by leading vendors of digital home equipment, is a nonprofit trade organization to discuss key specifications and standards of residential gateways and home networking diagnostics.

The two main theme of HGI are connectivity and service enabling, with specific reference to smart home scenarios. HGI appears to provide use cases, requirements and reference architectures for the home automation industry.

e. Industrial Internet Consortium (IIC)

Founded: 2014

Website: http://www.iiconsortium.org

Members: 140 (Intel, Cisco, AT&T, GE, IBM, Microsoft, HP, Samsung, NEC, Bosch, Dell, Siemens, Schneider Electric, etc.)

The IIC is a newly formed, nonprofit consortium to bring together the organizations and technologies necessary to accelerate growth of the Industrial Internet by identifying, assembling and promoting best practices.

The goal of the IIC is to influence the global development standards process for internet and industrial systems and make Big Data more accessible through the use of interconnected machines and the people who use them (and not directly create standards or open source implementations for IoT).

The IIC is managed by the Object Management Group, another nonprofit technology standards consortium dedicated to computer industry.

f. Internet Engineering Task Force (IETF)

Founded: 1986

Website: http://www.ietf.org

Members: NA

The Internet Engineering Task Force (IETF) develops and promotes voluntary standards to maintain and improve the usability and interoperability of the Internet, in particular the ones that comprise the Internet protocol suite (TCP/IP). It is an open standards organization, with no formal membership or membership requirements.

The IETF started out as an activity supported by the US Federal Government, but since 1993 it has operated as a standards development function under the auspices of the Internet Society, an international membership-based nonprofit organization.

IETF has published a number of standards for IoT developers, including *CoAP* for application messaging, *DTLS* for device security and *6LoWPAN* for network communication.

IETF has four active working groups related the adaptation of the protocols to the need of low power (or constrained) devices (Constrained RESTful Environments, IPv6 over Networks of Resource-constrained Nodes, Routing Over Low power and Lossy networks, Authentication and Authorization for Constrained Environments).

g. Internet Protocol for Smart Objects (IPSO) Alliance

Founded: 2008

Website: http://www.ipso-alliance.org

Members: 46 (ARM, Atmel, Bosch, EDF, Ericsson, Freescale, etc.)

The IPSO Alliance is a global, nonprofit forum seeking to establish the Internet Protocol as the basis for the connection of Smart Objects in industrial applications.

The IPSO Alliance was one of the first standards groups to operate in the IoT industry and works to complement the projects of other institutions, such as the Internet Engineering Task Force (IETF), the Institute of Electrical and Electronics Engineers (IEEE), the European Telecommunication Standard Institute (ETSI), and the International Society of Automation (ISA). For example, IPv4, IPv6, and 6LoWPAN were all developed within IETF, and the role of the IPSO Alliance was to ensure how they are used, deployed and provided to potential users.

h. M2M Alliance

Founded: 2007

Website: http://www.m2m-alliance.com

Members: 72

The M2M Alliance represents the interests of the M2M industry by creating better general conditions for attractive and lucrative M2M solutions through the constant interaction among industry, researchers and policy makers. According to the M2M Alliance promoting uniform and cross-industry standards represents a fundamental for creating new products and innovations. As an independent industry association the M2M Alliance supports its members in the growing M2M market as an ideal partner in the fields of hardware, software, integration and consulting.

It has a German focus and its main deliverable is the annual M2M Summit in Dusseldorf.

i. Machinery Information Management Open System Alliance (MIMOSA)

Founded: 1994

Website: http://www.mimosa.org

Members: 29 (IBM, Invensys, Rockwell, BP, Chevron, Raytheon, Aveva, etc.)

MIMOSA is a nonprofit trade association dedicated to developing and encouraging the adoption of open information standards for Operations and Maintenance in manufacturing, fleet, and facility environments. MIMOSA members come from process and discrete manufacturing companies, facility management companies, military organizations, capital equipment OEMs, and suppliers of asset management software systems.

MIMOSA identifies three fundamental parts: Business Management Applications (Open Systems Architecture for Enterprise Application Integration, i.e., OSA-EAI), Condition-based Management (Open Systems Architecture for Condition-Based Maintenance, i.e., OSA-CBM), and Reliability Centered Management.

The core of the MIMOSA development activity is the MIMOSA CRIS (Common Relational Information Schema) that defines a relational database schema for machinery maintenance information. The CRIS represents a static view of the data produced and managed by a CBM system.

j. Object Management Group (OMG)

Founded: 1989

Website: http://www.omg.org

Members: 287 (IBM, Apple, Sun, HP, etc.)

The Object Management Group (OMG) is an international, open membership, nonprofit computer industry standards consortium. OMG Task Forces develop enterprise integration standards for a wide range of technologies and industries, and set out to create the first Common Object Request Broker Architecture (CORBA) standard (which appeared in 1991) and Data Distribution Service for real-time systems (DDS).

OMG maintains liaison relationships with dozens of other organizations including ISO which publishes many OMG standards.

k. oneM2M

Founded: 2012

Website: http://www.onem2m.org

Members: 195 (ARIB, ATIS, CCSA, ETSI, TIA, TTA, TTC, etc.)

Seven of the worlds leading regional ICT standards bodies have come together to create a new global organization (i.e., oneM2M) with aim to develop a scalable and interoperable standard for communications of devices and services used in M2M and IoT applications that can effectively operate on a worldwide scale. Currently, oneM2M is working on M2M system standardization (e.g., CoAP, HTTP binding).

l. Open Interconnect Consortium (OIC)

Founded: 2014

Website: http://www.openinterconnect.org

Members: 6 (Intel, Atmel, Dell, Broadcom, Samsung, Wind River)

The OIC is a consortium, announced only a week before Thread, focused to deliver a standard specification, an open source implementation, and a certification program for wirelessly connecting device ensuring interoperability of IoT regardless of form factor, operating system, service provider or transport technology. The OIC intends to take the lead in establishing a standard for interoperability across multiple vertical markets and use cases.

The Open Interconnect Consortium already released IoTivity that promises to be an open source framework for device middleware, aiming to enable automatic device identification to facilitate networking and communication. IoTivity is a Linux Foundation collaborative project, and is therefore governed by an independent steering committee that liaises with the OIC. It is an open source project currently accepting contributions and this choice caused Broadcom to leave the OIC back in October 2014.

m. Open Mobile Alliance (OMA)

Founded: 2002

Website: http://openmobilealliance.org

Members: 96 (AT&T, Intel, Microsoft, Qualcomm, Motorola, etc.)

OMA was formed by the world's leading mobile operators, device and network suppliers, information technology companies and content and service providers. OMA is a nonprofit organization that delivers open specifications for creating interoperable services that work across all geographical boundaries, on any bearer network. OMA's specifications support the billions of new and existing fixed and mobile terminals across a variety of mobile networks, including traditional cellular operator networks and emerging networks supporting machine-to-machine device communication.

OMA has recently published a standard for IoT device management called Lightweight M2M (LWM2M) that is fast becoming the global industry standard for M2M.

The OMA is a member of oneM2M.

n. Organization for the Advancement of Structured Information Standards (OASIS)

Founded: 1993

Website: http://www.oasis-open.org

Members: 291 (IBM, Microsoft, ISIS Papyrus, Primeton, SAP, Cisco, Dell, NEC, PTC, Alfresco, Adobe, etc.)

OASIS is a nonprofit consortium that drives the development, convergence and adoption of open standards for the global information society.

OASIS promotes industry consensus and produces open standards for security, Internet of Things, cloud computing, energy, content technologies, emergency management, and other areas.

OASIS has recently published the MQ Telemetry Transport (MQTT) specification. Most IoT middleware providers now support MQTT and hardware suppliers are starting to include it. Eclipse Paho and Eclipse Mosquitto, which implement MQTT, are active and dynamic projects.

o. Thread Group

Founded: 2014

Website: http://www.threadgroup.org

Members: 50 (Nest, Samsung, ARM, Freescale, Silicon Labs, Big Ass Fans, Yale)

Thread was created by Google's Nest with the goal to develop the very best way to connect and control products in the home and is a marketing education group that offers product certification to ensure security and interoperability of products.

Even if the Thread Group claims not to be another standards organization, they have identified the necessity for a low-power, mesh network in addition to Wi-Fi to be used for connecting the objects. The Thread Group has now started to work on an IPv6 networking protocol built on open standards, designed for low-power 802.15.4 mesh networks, to provide a solution to this demand.

The Thread Group is now open to other companies and to anyone who wants to join. Thread-certified products (e.g., Nest thermostats to control home ceiling fans from Big Ass Fans) are already available on the market.

p. Standards Developing Organizations

In parallel, recognizing the value of IoT, the major Standards Developing Organizations (SDOs) both at national and international levels are creating specific working groups.

The IEEE is trying to build a global architecture for the Internet of Things, involving various industries and technologies. The working group P2413 Standard for an

Architectural Framework for the Internet of Things (IoT) started work on in July 2014, with 23 vendors and organizations represented, including Cisco, Huawei, GE, Oracle, Qualcomm and the ZigBee Alliance. The working group P2413 recognizes the evolving transformational integration and convergence across technology and application domains.
The architectural framework defined by P2413 will promote cross-domain interaction, aid system interoperability and functional compatibility. The adoption of a unified approach to the development of IoT systems aims to reduce current industry fragmentation and create a critical mass of multi-stakeholder activities around the world. The plan of the P2413 also provides a coordination with other organizations, including the European Telecommunications Standards Institute (ETSI), International Organization for Standardization (ISO) and the group oneM2M.
In addition, a joint technical committee of ISO end IEC is managing a special working group ISO/IEC JTC1/SWG 5 which supports and facilitates the development of standards for Internet of Things (IoT). This special working group aims at consolidation of works of internal agendas and external bodies rather than at publishing standards and identifies current and future trends and needs in IoT.
Finally, the Global Standards Initiative on Internet of Things (IoT-GSI) promotes a unified approach in ITU-T for development of technical standards (so-called Recommendations) enabling the Internet of Things on a global scale. ITU-T Recommendations developed under the IoT-GSI will enable worldwide service providers to offer the wide range of services expected by this technology. IoT-GSI also aims to act as an umbrella for IoT standards development worldwide.

In conclusion, there is not going to be one standard that will prevail in the IoT in the short-term. ICT standardization is a highly decentralized activity and IoT is now just too fast and complex. Probably we also expect more consortiums will be started in the future.

We will have to consider multiple scenarios. It seems to be something inevitable in a rapidly evolving industry. A possible way out of this chaotic situation would be to standardize the middleware and the XML-based data representation from across-industry organizations such as World Wide Web Consortium (W3C), Organization for the Advancement of Structured Information Standards (OASIS), and others (Zhou, 2012). In this context the middleware must be seen as the software layer between the hardware or operating system and applications levels (Bandyopadhyay et al., 2011). Another possible way out is a two-pronged approach, i.e., de facto standards, working groups, and recommended practices allowing products to be brought to market quickly; along with a longer-term effort to codify these practices into standards (DaCosta, 2013).

1.3. Protocols for the Industrial Internet of Things

In IoT and IIoT what connects the *Things* are the communication *Protocols*. So it is not possible to analyze IoT and IIoT applications without covering the communication protocols. In order to exploit the full potential of IoT paradigm the interconnected devices need to communicate using IP technology. In fact, IP architecture provides interoperability with existing networks, applications, and services. Besides, the IP stack is open, versatile, scalable, stable, and with a small footprint (Vasseur, and Dunkels, 2010).

At the network/transport level, in Open Systems Interconnection model (OSI) terms, there is a widespread adoption of the Internet protocol stack (i.e., TCP/IP). TCP/IP provides the identification, location, and routing that are core components of the IIoT architecture. On top of TCP/IP, the components of distributed applications have to decide how to communicate, i.e., *which application protocol* to adopt. In M2M and IIoT research areas, several protocols are thriving to become a de facto standard, and it could be difficult to identify the best solution for a given application domain. Lightweight protocols that don't require extensive use of CPU resources, i.e., MQTT, some scripting languages (such as

Python) are the preferable choices used by IIoT applications. A fundamental aspect of these protocols is whether it is Web-based like CoAP or application-based such as with AMQP. These aspects have fundamental effect on the environment, performance and tools available for developers.

Finally, it is worth mentioning that in some cases it may make sense to create a system that uses proprietary protocols. This may seem not to be the best choice but is a common scenario particularly as a system evolves and there is need to implement new applications. Moreover, some manufacturers prefer proprietary solutions for specific reasons (e.g., security issues). In conclusion, the challenge is to identify the manufacturing industry standard that will *enable Internet connectivity to manufacturing equipment*. The following sections describe some example protocols in more detail.

1.3.1. MQ Telemetry Transport (MQTT)

IBM developed WebSphere MQ as a message-based backbone mainly for Enterprise Application Integration. MQ Telemetry Transport (formerly Message Queue Telemetry Transport) was designed to permit WebSphere MQ to talk with small (smart) devices at the edge of the network. MQTT (http://mqtt.org) is an open-sourced protocol for passing messages between multiple clients through a central broker, i.e., a protocol for collecting device data and communicating it to servers (D2S). It was designed to be simple and easy to implement. The protocol is light-weight and therefore well suited for constrained environments, for example where the *network traffic is expensive*, has low bandwidth or is unreliable, or on *embedded devices* with limited processor or memory resources. Because the protocol was designed to be simple, users must decide whether it is too simple and susceptible to potential hacking.

MQTT is a protocol adopted in several platforms, i.e., there are MQTT server (i.e., broker) in WebSphere, Mosquitto, RabbitMQ, etc. and clients in main languages.

1.3.2. Constrained Applications Protocol (CoAP)

CoAP (http://coap.technology) is an Internet-based client/server model document transfer protocol similar to HTTP but designed for constrained devices, i.e., small devices with limited CPU, memory, and power resources often used as sensors/actuators, e.g., *Wireless Sensor Networks* (WSN).

CoAP is a low-overhead request/response protocol that also supports discovery of services/resources.

CoAP utilizes User Datagram Protocol (UDP) communications with support for multicast addressing. It is inspired by (and compatible with) the HTTP protocol and ReST architectures (CoAP is a specialized Web transfer protocol). UDP may be easier to implement in microcontrollers than TCP, but the security tools used for TCP (SSL/TLS) are not available in UDP. Datagram Transport Layer Security (DTLS) can be used instead.

CoAP is often used in conjunction with WSNs implementing the IETF's emerging IPv6 over Low Power Wireless Personal Area Networks (6LoWPAN) standard.

There are several platforms with Client/Server CoAP implementations, in many programming language (C, C++, Java, etc.).

1.3.3. Advanced Message Queuing Protocol (AMQP)

AMQP (http://www.amqp.org) is an application layer message-centric brokered protocol originally developed at JPMorgan Chase in London with the objective of replacing proprietary and non-interoperable messaging systems for the integration of enterprise IT components (enterprise message bus). The key features of AMQP are message orientation, queuing, routing (including point-to-point and publish-and-subscribe), reliability and security.

AMQP protocol can therefore be considered a queuing system designed to connect servers to each other (S2S). AMQP has a richer semantic (than MQTT), but is also heavier (for instance the broker is much more complex).

Most AMQP implementations, e.g., Apache focus on providing functions such as transactions, queuing, message distribution, security, management, clustering, federation, and heterogeneous multi-platform support, most of which are not essential in the M2M/IIoT scenario.

1.3.4. Data Distribution Service (DDS)

DDS (http://portals.omg.org/dds) is a data-centric middleware language designed to enable scalable, real-time, dependable, high performance and interoperable *publish-subscribe messaging*. In addition to the AMQP semantics, DDS also supports complex communication patterns and Quality of service (QoS).

The main purpose of DDS is is to connect devices to other devices (D2D).

DDS is proposed in several environments, e.g., industrial automation (also for SCADA communications), financial applications, and even military environments, where performance and reliability are essential.

Many DDS implementations are proprietary and rely on a full-fledged CORBA system (DDS was part of the CORBA specification until 2012).

DDS is both language and OS independent. The DCPS APIs have been implemented in a range of different programming languages including Ada, C, C++, C#, Java, JavaScript, Scala, Lua, and Ruby. Using standardized APIs helps ensure that DDS applications can be ported easily between different vendor's implementations. While DDS vendors are increasingly focused on mobile applications, e.g., for Android, their platforms might be too heavy for M2M/IIoT applications. DDS Security specification is still pending.

DSS is available commercially and a version of it has have been made available as open.

1.4. Summary and Concluding Remarks

The messaging technologies discussed above are the main solutions that can be used to connect machines and devices (e.g., sensors, mobile devices, single board computers, servers) in a distributed network (e.g. LAN or WAN) via a range of wired and wireless communication technologies (e.g. Ethernet, Wi-Fi, Zigbee, Bluetooth, GSM, GPRS, UMTS 3G, LTE 4G). There is also a number of other emerging key messaging technologies that will support the next generation of M2M and IIoT applications and foster large scale e-Maintenance platform. Choosing between these alternative solutions is usually problematic. The identification of the best suited messaging technologies must be based on an analysis of both the architecture and the message/data sharing requirements for each target system. For example, although there are obviously overlaps, turning on the light with a switch (where CoAP could be seen as the right choice) is different from generating power (where it is preferable to use DDS), monitoring the transmission lines (where MQTT is the proper choice), or analyzing the power usage at the server room (where AMQP handles the case well).

Chapter 2

Large-Scale e-Maintenance

2.1. Introduction

e-Maintenance is a recently emerged discipline that relies on the integration of information and communication technologies (ICT) into the maintenance processes in order to enable the remote management of industrial equipment and to support proactive decision-making and efficient maintenance operation planning (Muller, Crespo-Màrquez, and Iung, 2008). In this context, e-Maintenance represents a major evolution of traditional maintenance practices that expresses the synthesis of two major trends in today's society, that is, the growing importance of maintenance as a key technology and the rapid development of ICT. e-Maintenance integrates and synchronizes the various maintenance and reliability applications to gather and deliver asset information where it is needed, thus improving adaptation and resiliency (Campos, 2009).

So far, e-Maintenance solutions have been developed mostly for large and expensive machinery, for instance, in the energy industry and in the heavy one because of the high costs of remote monitoring technology (Kunze, 2003; Garcia, Sanz-Bobi, and del Pico, 2006). However, the cost for the implementation of e-Maintenance functions is significantly decreasing thanks to recent ICT developments, such as powerful components off the shelf (COTS) hardware, the emergence of Web technologies as the de facto standard for interoperable distributed applications, the evolution of data-intensive real-time processing techniques that enable the highlighting of nontrivial (cor)relations among data, and the widespread availability of (relatively) high-bandwidth Internet-based communications.

These driving factors are now fostering the evolution of e-Maintenance platforms from *large plants* to *large-scale* machine installations. This enables the application of e-Maintenance to *household and similar appliances*, that is, low-cost automated machines, such as those designed to be operated by untrained personnel in homes, shops, restaurants, offices, warehouses, light industries, or farms. Usually, household appliances and similar machines incorporate motors, compressors, heating elements, CPUs, or all of the above. Because there is no on-site technical support available, remote monitoring and diagnosis are crucial to organize and schedule timely assistance operations in order to improve the after-sales assistance of these machines.

State-of-the-art large-scale e-Maintenance platforms can provide an array of remote assistance services for the management of automated machines, such as *automated monitoring*, *self-diagnostics* and *prognostics*, and *remote management*.

The evolution of large scale e-Maintenance platforms is also likely to have a major impact on the design of next-generation automated machines, leading the to realization of adaptive machine-installed control software (and firmware) *that enables a dynamic change in the production workflow* in response to component wearing, to changes in environmental conditions, or to the need to change the result of the manufacturing processes (for instance, a food processing machine can produce either ice cream, frozen desserts, or chocolate by simply changing its internal thermal cycle, i.e., temperature vs. time).

In addition, the maintenance data collected by large-scale e-Maintenance platforms will surely become an invaluable asset for the enterprise, thus leading to the development of innovations, efficient management strategies, and new business models.

The work presented in this dissertation emerge from the author's experience in building an innovative e-Maintenance platform, called Teorema, for the management of the ice cream machines produced by Carpigiani Group (one of the two leading industries in the worldwide market). Teorema is currently enabling the remote assistance of more than 5500 machines distributed all over the world, thus allowing for significant savings in maintenance operations, and represents a disruptive innovation in after-sales services within the ice cream machine market.

2.2. From Maintenance to e-Maintenance

Maintenance is the field of management that consists of skills, techniques, methods, and theories that aim at developing technical and organizational solutions to ensure the correct performance of the required functions in industrial equipment.

Maintenance is an essential process for the preservation of large assets, such as factories, power plants, vehicles, and buildings, as well as of smaller items, such as vending machines, appliances, and commercial equipment in working conditions—also in terms of energy saving, environmental impact, and safety.

In recent years, engineers and researchers have developed different organizational approaches to improve the quality of maintenance, such as total productive maintenance (TPM, Nakajima, 1988), reliability-centered maintenance (RCM, Moubray, 1997), and condition-based maintenance (CBM, Crespo-Màrquez, 2007). These methods have been implemented in industry and process plants with excellent results.

TPM consists of a range of methods developed in Japan in recent times, which aim at maximizing plant and equipment efficiency. TPM is closely related to total quality management and can be seen as an application of a continuous improvement plan-do-check-act cycle, otherwise known as a Deming cycle. TPM is based on an organizational model that involves machine operators in the maintenance process. More specifically, TPM requires operators to take over some of the day-to-day maintenance tasks, for example, cleaning, lubricating, tightening, and reporting of changes observed in the operation of the equipment.

RCM follows a different approach—less pervasive than that of TPM. The goal of RCM is to identify the minimum levels of maintenance required to keep machinery in safe operating conditions as well as to implement the proper processes and policies that enable the enforcement of these maintenance standards. As a result, RCM can be defined as a technology-centric concept, in contrast to the TPM that instead represents a human-centric technique. In fact, RCM is a highly structured method for the planning of maintenance, originating from a critical analysis of maintenance requirements.

CBM marks the key conceptual shift from *preventive maintenance* strategies, such as TPM and RCM, to *predictive maintenance* ones. In fact, by leveraging condition-monitoring technologies that measure physical (temperature, pressure, viscosity, vibrations, etc.) and functional working parameters in order to determine the current status of a piece of equipment, CBM enables the planning of maintenance interventions when deterioration in equipment condition occurs. As a result, CBM allows the reduction of the number of maintenance interventions and the consumption of spare parts with respect to preventive maintenance strategies, resulting in significant savings.

However, rapid changes and developments in ICT have recently brought disruptive innovations in industrial process management and maintenance, opening the way to the implementation of *proactive maintenance* (Voisin et al., 2010). Proactive maintenance differs from predictive maintenance as it aims to implement more sophisticated monitoring techniques that go beyond the detection of equipment (or component) degradation or machine wear after they have taken place. In fact, proactive maintenance focuses on the early detection and the correction of problems that might lead to equipment malfunctions, thus enabling easier prevention of failures.

The introduction of ICT in maintenance has given rise to e-Maintenance, a recently emerged discipline that aims to enable the remote management of industrial equipment and the planning of efficient maintenance operation (Muller, Crespo-Màrquez, and Iung, 2008), which represents the key component for the realization of proactive maintenance in the digital factory (Chryssolouris et al., 2009). e-Maintenance platforms augment the set of maintenance tools available to technical support personnel to enable the restructuring of such maintenance processes as monitoring, diagnosis, and prognosis in order to fully reap the benefits provided by proactive maintenance (Campos, 2009; Lee et al., 2006; Levrat, Iung, and Crespo-Màrquez, 2008).

Figure 1 depicts the relations between the different maintenance policies and strategies.

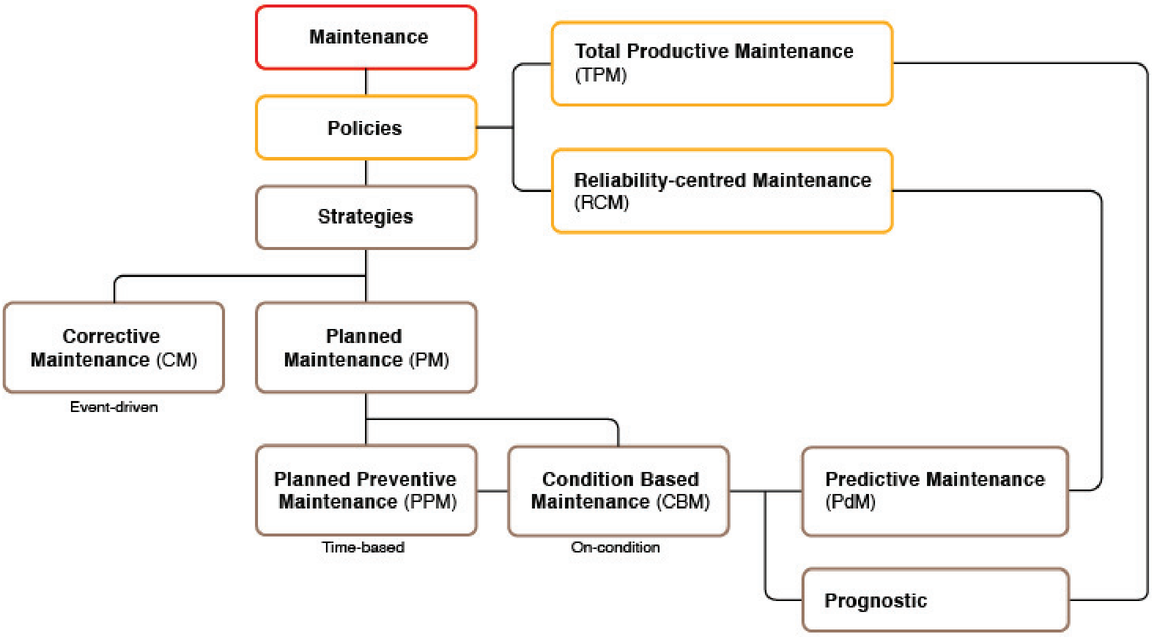


Figure 1 – Structure of Maintenance

2.3. e-Maintenance: From Large Plants to Large Scale

So far, most of the research work in e-Maintenance has focused on large plant machinery, in particular in the oil and gas industry (Wan Mahmood et al., 2011), in the energy industry (Kunze, 2003), and in the heavy one (Garcia, Sanz-Bobi, and del Pico, 2006). The size and cost of those machines justify and encourage the development of sophisticated e-Maintenance solutions to improve efficiency. To implement 24/7 remote monitoring, e-Maintenance solutions for large and critical machinery usually adopt expensive and often proprietary supervisory control and data acquisition (SCADA) technologies that leverage relatively high-bandwidth ad hoc communication stacks, for example, private radio over UHF or unlicensed bands. SCADA-based e-Maintenance solutions usually leverage dedicated hardware and complex software and communication technologies to realize the remote monitoring functions and are difficult to deploy on a large scale as they require a significant effort to enable the interoperability between equipment from different vendors (Brunello, 2003) and to realize a proper communications infrastructure (Yang, Barria, and Green, 2011). The cost of SCADA-based e-Maintenance solutions make them unsuited for adoption in the case of low-cost, mass-produced equipment, such as household and similar appliances.

Modern household and similar appliances are often complex and high-duty systems, which perform critical operations, for example, in food-processing applications, and have a long (often more than a decade) lifecycle. As a result, notwithstanding the quality of their components and of their manufacturing processes, during their lifetimes, these machines typically require numerous maintenance interventions. These machines can be deployed in extremely heterogeneous environments, such as restaurants, shops, farms, and homes, where they operate unattended with no on-site technical support personnel available. Household and similar appliances are typically managed with a reactive maintenance strategy, based on on-site maintenance interventions scheduled according to a combination of "run to failure" and periodic maintenance policies. Because the technical personnel working in the field have little or no knowledge about the problem before performing an on-site machine diagnostics procedure, they sometimes might not carry the spare part needed for fixing the specific component that has failed (first-time-fix, i.e., FTF). Thus, technicians have to delay the problem resolution to a second on-site intervention, leading to high machine downtime and revenue losses or to substituting larger parts of the machine, leading to an excess consumption of spare parts. As a result, reactive maintenance strategies for household and similar appliances usually have very high costs and low level of customer service quality.

These considerations suggest that e-Maintenance platforms could bring significant benefits for the management of large-scale installations of household and similar appliances, enabling considerably more effective and more efficient maintenance processes based on remote operations and proactive interventions, especially with regards to the efficient scheduling of human resources that have to perform both remote and on-site maintenance interventions.

2.4. ICT Building Blocks for e-Maintenance

Several ICT trends are fostering the adoption of e-Maintenance for large-scale installations of automated machines, enabling the implementation of sophisticated machine-installed monitoring functions, which were once only possible for large and costly equipment (Lazzarini et al., 2013; Jantunen et al., 2011).

Commercial off-the-shelf (COTS) hardware is becoming more and more powerful and affordable, thanks to the huge demand and the fierce competition in the mobile and embedded devices markets.

Web-based software technology is allowing the realization of sophisticated and highly interoperable communication infrastructures for distributed applications, leveraging open source software that has low barriers to entry, no license costs, and rapid development process.

Advances in distributed data analysis techniques enable the real-time analysis of large amounts of data, thus allowing the implementation of sophisticated anomaly detection techniques to proactively identify problems within machines. Finally, the formidable improvements in Internet technology now provide several solutions, for example, Internet and mobile connectivity, which can significantly facilitate the remote access to appliances, with low costs, especially with regard to data traffic.

2.4.1. Evolution of COTS Hardware

Low-cost and low-power hardware platforms have seen a significant development recently. Following a large demand for affordable CPUs in the smartphone and tablet and in the home automation markets, relatively high-performance COTS microprocessors are now available at prices that are becoming affordable for large scale adoption.

This increases the computational capabilities available for machine-installed (self-)monitoring equipment while keeping the hardware costs reasonably low in order not to excessively increase appliance production costs.

As an example of the kind of processing power-to-cost ratio that is currently available on the market, we mention the Raspberry Pi initiative (http://www.raspberrypi.org), which sells ARM-based credit card–sized computers running a full-fledged version of the Linux operating system for $35 with the purpose of supporting and fostering software-development education. The well-known Arduino initiative, that develops and sells a family of Open Source micro-controllers for electronic prototyping with a starting price of about $20, has also recently released significant developments, such as the powerful Pentium-based Intel Galileo board and the next-generation Arduino TRE board, which integrates an ARM-based processor running Linux on the micro-controller board.

Let us notice that the availability of cheap ARM-based solutions represents a significant advance for embedded hardware applications. In fact, these enable the running of full-fledged operating systems, such as Linux with the Xenomai (http://www.xenomai.org) extension, that are also capable of delivering (soft) real-time guarantees, thereby significantly facilitating the operation of porting existing software, for example, anomaly-detection algorithms. At the same time, this provides developers with tools and languages that provide high-level programming abstractions that are typical of traditional (desktop and Web) applications and enables COTS software component reuse. This considerably increases productivity and, at the same time, lowers the barriers to entry in the embedded applications market.

2.4.2. Web-Based Software Technologies

The development of Web-based software technologies is also a huge driving factor behind the realization of large-scale e-Maintenance platforms. In fact, the interoperability standards developed so far in the industrial computer applications market, such as the Open Systems Architecture for Condition-Based Maintenance, i.e., OSA-CBM project (Lebold and Thurston, 2001), have not yet delivered protocols and service interfaces that are well suited for the integration of systems on a large scale.

These application scenarios could benefit from the experience in the realization of truly open and easily integrated systems developed in the Internet ecosystem and Web applications market. Web technologies are the de facto standard in the realization of interoperable distributed applications. Web services are the basis for service-oriented architectures (SOA) that enable easily dealing with heterogeneity in the hardware and software platforms as well as the realization of complex services through the composition of simpler ones (Zhu, 2003).

In addition, Web services are evolving beyond MQTT-based solutions to encompass CoAP-based solutions that are easier to integrate and significantly more scalable and available on low-power devices. ReST-based solutions for the development of distributed systems originally available only for PC-class machines are now available for embedded devices (Belqasmi, Glitho, and Fu, 2011).

The modern Web-based software-development technologies for distributed and embedded applications represent an ideal basis for the realization of complex management functions, such as diagnostics and prognostics.

2.4.3. Data-Intensive Real-Time Processing Techniques

The recent evolution of data-intensive real-time processing techniques enables the processing of large quantities of data and to highlight nontrivial (cor)relations among them. This is of critical importance for large-scale e-Maintenance platforms, which have to process concurrently the maintenance data received from a high number of automated machines for diagnostics and prognostics purposes.

In these kinds of applications, the large scale of data to process forces the development of methods that realize smart and efficient data storage and data processing. In addition, there is the need to adopt scalable data-analysis solutions that can perform diagnostic and prognostic procedures for each automated machine, comparing the maintenance data with those coming from machines operating in similar conditions (product life cycle history, installation type, environmental and climatic conditions, operating conditions, intended use, etc.).

To this end, e-Maintenance platforms can leverage modern data-fusion techniques developed in sensor network applications (Mitchell, 2012) and can take advantage of modern data-mining techniques, which enable the detection of nontrivial associations in large data sets (Reshef et al., 2011) and the performance of queries on data collected from multiple sources of both a structured as well as an unstructured type (Freitas et al., 2012). Finally, for the scalable execution of computationally intensive data-processing tasks, e-Maintenance platforms can exploit the elastic and parallel computations provided by cloud-based systems (Sakr et al., 2011).

Many of the above-mentioned techniques and tools were developed in the context of the Big Data application field (Alexander, Hoisie, and Szalay, 2011), which has recently attracted a lot of interest from both the industry and the academy.

The term “Big Data” encompasses all those issues that arise from the analysis of extremely large quantities of data, typically found in many modern applications, such as in the business intelligence and in the scientific computing fields (Bryant, 2011; Ahrens et al., 2011).

2.4.4. Ubiquitous and Large-Bandwidth Internet-Based Connectivity

The widespread diffusion of Internet connectivity significantly facilitates the deployment of large-scale e-Maintenance platforms, which need to provide remote access to a high number of machines installed in a wide array of locations.

To this end, it is often possible to leverage a preexisting Internet connectivity infrastructure at the customer's premises. This enables the easy provision of remote access to appliances, usually with high-bandwidth connectivity and at negligible costs. However, some locations might not have wired access to the Internet or might not allow the exploitation of it. For instance, in some situations, it may be advisable not to convey maintenance data over the LAN installed at the customer's premises for security reasons. In those cases, mobile connectivity can be exploited instead.

Let us notice that, in the context of mobile communications, e-Maintenance solutions can take advantage of commercial offers originally designed for the machine-to-machine (M2M) market (Katusic et al., 2012).

2.5. e-Maintenance Platforms for Large Scale

The realization of e-Maintenance platforms for this class of machines presents some peculiar challenges because they have to deal with large-scale installations, heterogeneous machines, and network connectivity and discontinuous transmission of maintenance data.

More specifically, e-Maintenance platforms in charge of controlling automated machines on a large scale need to provide a basic set of functions to support the main operations required. First of all, these machines usually operate unattended with no on-site technical support personnel available. It is therefore of utmost importance to minimize failures by adopting proactive maintenance practices. In fact, the ability to detect upcoming failures early is essential in order to reduce operation downtimes, maintenance costs, and safety hazards of appliances and therefore to preserve revenues.

To achieve this goal, a large-scale e-Maintenance platform typically implements four main functions: *automated monitoring*, *self-diagnostics*, *prognostics*, and *remote management* (Muller, Crespo-Màrquez, and Iung, 2008). In addition, many of these platforms are capable of integrating with the enterprise information systems to generate value-added services, for instance, by providing the machines' production data to their customers (Lazzarini et al., 2013). Finally, large-scale e-Maintenance platforms are typically designed to address the security concerns raised by the remote management of machines.

2.5.1. Automated Monitoring

The automated monitoring function is very important for any large-scale e-Maintenance platform. In fact, the availability of highly informative maintenance data is essential in order to accurately capture the current state of the machines, to enable the scheduling of field service interventions only when really needed, and to realize periodical and/or on-demand (when applicable or appropriate) reporting of the operational state (including incipient and conditional failure notifications).

The proactive management of household and similar appliances demands the monitoring, as closely and as continuously as possible, of a potentially large number of variables. In fact, the monitoring should not only consider the variables that enable the evaluation of the degree of equipment degradation, but also those that might enable the detection of the presence of hidden problems before significant degradation takes place.

The automated monitoring function is in charge of collecting maintenance data from the machines, of predisposing them for further processing through the application of aggregation and manipulation procedures, and of enabling their analysis by both automated diagnostic and prognostic components as well as technical support personnel through a dedicated management interface.

More specifically, an e-Maintenance platform typically implements an interface that enables the synoptic visualization of the machine state (programming table, firmware version, analysis of alarms, diagnostics of failures, events, machine access log), adopting the proper graphical and time-varying representation for critical metrics (pressure values, temperature values, etc.) and enabling the monitoring of values in real time for a live monitoring session.

In addition, e-Maintenance platforms usually provide advanced reporting mechanisms that automate and facilitate the machine-monitoring processes, for example, sending to each technical support employee a daily report about the state of the machines under his responsibility, etc.

2.5.2. Self-Diagnostics and Prognostics

The self-diagnostic and prognostic functions of large-scale e-Maintenance platforms realize both the automated fault diagnosis (usually implemented directly on the machine side) as well as the comparative analysis of data collected from all the machines in order to detect deviations, anomalous operations, or conditions favorable to the fault.

The early detection of anomalies and the forecast of machinery health based on the collected maintenance data are essential to enable proactive maintenance as they represent the best way to reduce both machine downtime and maintenance costs.

As a result, the implementation of diagnostic and prognostic functions through online or just-in-time data analysis techniques that enable the real-time processing of the collected maintenance data is highly desirable.

Sophisticated prognostic techniques, together with ongoing advances in data-fusion techniques, and increasingly comprehensive databases of condition data hold promise for improved design of new machines and adaptive strategies of maintenance management in real-world scenarios.

It is also vital to perform timely notifications in case of unexpected behaviors detected in one ore more machines. To this end, an e-Maintenance platform typically exploits a messaging infrastructure to notify the technician who is in charge of its maintenance in a timely manner.

2.5.3. Remote Management

Large-scale e-Maintenance platforms also provide a remote management function, which is essential in order to implement a proactive maintenance strategy. In fact, this function allows the establishment of on-demand live monitoring and diagnostic sessions to help technicians quickly identify the cause of deviations and abnormal operations on the machines under their responsibility and in deciding how to proceed for their resolution. In addition, these platforms typically provide a remote reconfiguration function to enable the automation of firmware and configuration upgrades for the machines.

Generally, the remote management functions provided by large-scale e-Maintenance platforms are specifically designed to allow non-intrusive measurements and analyze machine operation and production data in real time from different locations, thus supporting the repair companies and the staff personnel working on the field.

The remote management function enables not only the minimization of the time between the notification of failure and the beginning of the resulting service intervention, but also to plan and optimize the work time of the highly skilled specialist technicians, a critically important asset. Traditional maintenance strategies, exclusively based on on-site field service, would instead force expert technicians to spend most of their time traveling to and from the machine site locations within large geographic areas (Lazzarini et al., 2014; Holmberg et al., 2010).

2.5.4. Security

Finally, let us notice that the remote management of machines raises significant security concerns. The recent Stuxnet malware attack, aiming to spy on and reprogram SCADA systems, has revealed once more the need for security (Chen, and Abu-Nimeh, 2011). Attacks on the Internet of Things (IoT) devices and machines will increase rapidly due to the growth in the number of connected products and the high value of data on them. In fact, maintenance data is an important industrial secret that has become the objective of a large (and growing) number of malicious attacks (Cai, Wang, and Yu, 2008). As a result, there is the need to protect confidentiality of data transfers between appliances and external monitoring stations. It is also necessary to prevent malicious access to the machines from the communication link to avoid the unauthorized changing of critical configuration parameters of the manufacturing process and, consequently, malfunctions (and potentially damage) of machines.

These considerations suggest the opportunity to adopt a secure-by-design software-engineering practice when realizing e-Maintenance platforms. Designing the system to be secure from the ground up is essential to hindering malicious attacks and to minimizing the impact of security vulnerabilities.

2.6. Other Remote Control Systems

For their peculiar characteristics, large-scale e-Maintenance platforms for automated appliances present significant similarities to smart utility networks (Sum et al., 2011). Smart utility networks are telemetry systems for the efficient management of energy and other utilities. While these systems share similar remote monitoring concerns with e-Maintenance platforms, research on smart utility networks focuses on the realization of wireless mesh networks enabling access to electronic devices. Instead, e-Maintenance platforms leverage a preexisting Internet connectivity infrastructure to provide access to remotely monitored machines and realize much more sophisticated services, such as the visualization of synoptical data and fault detection based on complex event analysis.

Some aspects of the Internet of Things scenario, such as crowd-sensing (Ganti, Fan, and Hui, 2011), are also similar to the monitoring function of e-Maintenance platforms for household and similar appliances. However, the Internet of Things focuses on the design and development of interoperable architectures that enable the collection of data from remote and extremely heterogeneous devices, usually through the realization of semantic service infrastructures (Pfisterer et al., 2011) or the deployment of machine-to-machine communication solutions on top of existing service platforms (Foschini et al., 2011). Instead, the e-Maintenance presents a set of different situations characterized by wide range of variations. For example, in some cases, operating with machines of a single vendor, e-Maintenance platforms have less strict device heterogeneity issues to address, but they have to deal with the coordination of maintenance data transmission, enabling users to choose which specific data to obtain and to receive it in real time. In addition, e-Maintenance platforms have to provide advanced distributed management functions, such as remote device reconfiguration and automated diagnostics and prognostics.

M2M communications represent a similar application to e-Maintenance, of which, to some extent, they could be considered a precursor. In fact, M2M aims at realizing distributed monitoring of devices, mostly leveraging on narrow-bandwidth GSM-based connectivity (Wu et al., 2011). The business opportunities in the M2M market that promised to enable the monitoring of electronic devices on a global scale leveraging GSM-based text-messaging solutions led many mobile operators to offer (relatively) cheap specialized data-only mobile contracts. Large scale e-Maintenance platforms can benefit from these offers to realize support for the remote monitoring and assistance of automated machines.

2.7. Evolution of Automated Machines

The considerations presented in previous sections enable the envisioning of a future scenario in which automated machines are connected through a monitoring network and a significant portion of maintenance interventions is performed remotely.

Automated machines would then become an adaptive and resilient system of interconnected, self-monitoring, and self-managing nodes.

2.7.1. Next-Generation Automated Machine Design

The application of e-Maintenance platforms for the large-scale remote monitoring and control of automated machines has a disruptive potential that could have a considerable impact on the way in which these machines are designed and maintained.

In fact, the design of automated machines could change significantly as a result of the integration of the (relatively) advanced ICT support required to realize e-Maintenance functions. It is possible to envision that in next-generation automated machines the software/firmware will not only be capable of controlling the transitions between the steps of the (preconfigured) production process workflow implemented by the machine, as in most of the machines currently produced, but that it will enable a much finer-grained control over the entire production processes. Next-generation machines will expose a much larger portion of their internal state to the control software/firmware and enable the software/firmware not only to change from one stage to the next of a preconfigured manufacturing workflow, but *to change the workflow itself*.

More specifically, the software components running in automated machines would have access to a much larger number of observable variables (or sensors) as well as to a larger number of controllable variables (or actuators). This would make it technically possible to switch to a different manufacturing process workflow, enabling the implementation of customer-specific processes (for instance, the customization of machines to implement specific thermal cycles required by signature recipes is very important in the ice cream machines market). Machines would therefore change from being *mildly controllable* to being *highly controllable*.

From the maintenance point of view, the fault detection and the fault recovery components might cooperate to enable the machines to autonomously switch to a less efficient but more robust one that would work with slightly deteriorated components in case of failures.

This would also represent a significant step toward *mass product customization* in the household and similar appliances market. The adoption of e-Maintenance solutions would foster the mass customization process, which is already reshaping some markets, such as the automotive one (Shamsuzzoha, and Helo, 2009), in the household and similar appliances market. In fact, e-Maintenance would enable manufacturers to produce identical machines for all the customers from the hardware point of view and to reconfigure their software and firmware on demand to implement the specific operations and the manufacturing processes required by each specific customer even at the single-machine level.

Another interesting evolution in automated machine design might be the possible *adoption of over-engineering and/or overprovisioning practices*. In fact, as e-Maintenance enables significantly cheaper after-sales assistance compared with traditional maintenance practices, for next-generation automated machines over-engineering design approaches, such as the inclusion of redundant components and the automated fallback to secondary components when primary ones fail, might be more economically convenient than a traditional design that would lead to forced costly downtime in case of failures.

Finally, let us notice that the software/firmware is becoming, more and more, the most valuable part of the machine. This is likely to raise even more the problem of preserving intellectual propriety in the near future.

2.7.2. Evolution of Maintenance Data-Processing Techniques

We can also envision an evolution of the maintenance data-analysis practices. More specifically, monitoring tools for next-generation automated machines will be able to analyze and process a larger array of data streams, of significantly different type and nature, through sophisticated anomaly-detection techniques. This will enable the extrapolation of nontrivial features and ultimately build a deeper knowledge of the machine internal state, which could help in developing sophisticated diagnostic and prognostic tools.

In fact, with the increase in computational and bandwidth capabilities and the availability of more and more sophisticated sensors, e-Maintenance platforms will be able to collect and analyze a significantly larger amount of maintenance data, which could be processed locally or at a central monitoring station according to the specific requirements of the monitoring process and of the data characteristics.

For instance, accelerometer sensors, which generate a large amount of data that is computationally very expensive to analyze, as a result, are currently mounted only on large plant machines, and they could be installed even on low-end automated machines to monitor internal processes. At the same time, it is possible to hypothesize that the evolution of sensors could enable significantly more sophisticated process-monitoring solutions, for example, based on biological sensor data acquisitions.

In addition to a larger amount of data, next-generation monitoring solutions will also have to deal with a larger number of data types to analyze. In fact, in most cases, the maintenance data currently analyzable consists of discrete data sets, also known as “value type” monitoring data in the maintenance literature (Jardine, Lin, and Banjevic, 2006), which essentially limits the monitorable quantities to temperature, pressure, etc. Discontinuous value-type data are significantly more difficult to handle than continuous data, for which a large number of well-tested and sophisticated tools (based, for instance, on Fourier analysis, wavelet analysis, time-series analysis, etc.) is available, and present some limits to the applicability of automated anomaly-detection mechanisms for real-time monitoring systems (Chandola, Banerjee, and Kumar, 2009).

In addition, many models of automated machines also record a list of events, including events related to the normal operating conditions of the machines, such as commands issued by the machine operators, completion of manufacturing process, etc., and transfer them to the central management station. This information, also known as "event data" in the maintenance literature, is used to realize higher-order detection of malfunctions or incorrect machine usage through complex event-correlation techniques (Martin-Flatin et al., 2007).

However, let us point out that the processing of these types of maintenance data, which requires the correlation between value type, continuous time-series, and event data streams, might require significant advances of data analysis techniques, thus calling for both theoretical and applied research.

2.7.3. Leveraging on e-Maintenance Data to Optimize Maintenance Processes

The data collected by e-Maintenance platforms will surely become a strategic asset for automated machine manufacturing enterprises, one that can be leveraged to guide and improve the maintenance processes.

In fact, the data collected from e-Maintenance platforms represents an invaluable source of information for the restructuring of maintenance processes toward more efficient practices. For instance, the analysis of e-Maintenance data could foster the adoption of efficient management practices for expert technical support personnel, which represents a critical issue when dealing with large installations of automated machines. In fact, despite the adoption of sophisticated practices, such as predictive maintenance (Mobley, 2002) and condition-based maintenance (Jardine, Lin, and Banjevic, 2006), the domain expertise of human personnel has always proved to be a critical asset to enable effective maintenance decisions (Prakash, 2006; Veldman, Klingenberg, and Wortmann, 2011). Unlike large machinery and factory production shop floors, household and similar appliances operate without the presence of technical support personnel, and their maintenance interventions typically require relatively long waiting times between malfunction notifications and the arrival of technicians on-site. It is therefore essential for e-Maintenance platforms to provide functions that allow the exploitation of, at best, the expertise of the most skilled technicians, for instance, by enabling the performing of assistance interventions remotely and the building and sharing of knowledge within maintenance teams (Ribeiro, Barata, and Silvério, 2008). These considerations also suggest the opportunity to exploit groupware tools to facilitate the collaboration between technicians in dealing with maintenance operations (Hedjazi, and Zidani, 2011).

2.7.4. Leveraging on e-Maintenance Data to Develop Servitization

The term servitization can be defined as the process of creating value by adding services to products (Vandermerwe, and Rada, 1988; Baines et al, 2009).

Servitization represents a fundamental business model shift in which products and services will be bundled together to form a single entity, a Product-Service System (PSS), and a new value. This shift begins with effective management and delivery of aftersales support services, then products bundled with warranty, and finally performance-based service contracts (Baines and Lightfoot, 2013).

The adoption of performance-based (i.e., *pay-per-use* or *power-by-the-hour*) contracting is becoming more and more common for capital-intensive products in the defense and air transport industries (Kim, Cohen, and Netessine, 2007) and also in the household and similar appliances market. In performance-based contracting, customers do not purchase a product but instead subscribe to a service that warrants the availability of the above-mentioned product (or an equivalent one) in fully operating conditions for a prearranged time interval. The service-oriented business models, of which performance-based contracting represents an important example, has proven to be a very effective way to align a company's offer to its customers' needs (Oliva, and Kallenberg, 2003). In this context, e-Maintenance solutions, which enable the efficient monitoring and servicing of machines for both management and production, might increase the attractiveness of performance-based contracting and help companies currently adopting a strictly product-based business model to proceed toward a more service-oriented offer. In the future, more and more, business models will be shaped by data and technology (Baines, and Lightfoot, 2013).

2.7.5. Leveraging on e-Maintenance Data to Improve Machine Design

In addition to the previous point, you must consider that the data collected by e-Maintenance platforms is also very important to help the evolutionary design of machine components. In fact, by analyzing monitoring data collected from machines, engineers can extrapolate invaluable information. This knowledge could significantly help during the design phase by enabling engineers to perform accurate assumptions about the operating conditions that components will have to face once deployed as well as to avoid over- or under-specifications of the components that could lead to higher manufacturing or maintenance costs.

This aspect is closely related to the concepts of *open innovation* and *user innovation*.

Open innovation is a paradigm that assumes that firms can use external sources of innovation such as customers, rival companies, and field data collection (Chesbrough, 2003).

User innovation refers to innovation by consumer users (individual end-users or user communities), rather than by producers or manufacturers (Bogers, and Bastian, 2010).

Open source is a natural way of innovation in the software industry and it is a paradigmatic example of open innovation, with open-source projects/communities act as innovation intermediaries (Von Hippel, 2001).

This two approach represents a paradigm shift with respect to innovation research and offers both inescapable challenges and opportunities.

Finally, large volumes of data from the field require statistical tools and data mining techniques in order to have information useful to modify the product design and/or the manufacturing processes.

2.7.6. Deployment Considerations

Along with the trend that is pushing the intelligent management substrate in the machines, one could envision future household and similar appliances that operate (semi-)autonomously, possibly evolving into groups that automatically deal with failures and anomalies. For instance, upon failure, machines could automatically re-dispatch suspended manufacturing processes to other similar machines within the same premises. Or, in case of anomalous readings, co-located machines could exchange maintenance data to assess if the anomaly is due to an environmental factor.

To support the realization of self-management functions at the machine group level, it is possible to think of deploying local monitoring stations that could perform the analysis of maintenance data from all the machines in the premises. This could facilitate maintenance data-analysis operations, which could comparatively evaluate data coming from machines operating in the same environmental conditions and make the e-Maintenance platform more resilient to network disconnections, thus effectively implementing self-managed systems that can withstand disconnections from the central remote operations management facility of their e-Maintenance platform.

2.8. Large-scale e-Maintenance to Provide Prognostics Tools for Machines

Networked and remotely monitored machines, and their data modelled and continually analyzed, make it possible to go beyond mere predictive maintenance to prognostics. Prognostics is defined as a systematic approach that can continuously track health degradation and predict the Remaining Useful Life (RUL) of a component, which is an important concept in decision making. Prognostics foresees the future performance of a component by assessing the extent of deviation or degradation of a system from its expected normal operating conditions.

Figure 1 depicts the relations between the prognostics and different maintenance policies and strategies.

Prognostics uses methods such as statistical analysis, pattern recognition techniques (Lopez, 2007; Wang et al., 2011), and Machine Learning to predict the Remaining Useful Life (RUL) of machine components (Schwabacher, and Goebel, 2007). The information produced by such tools allows companies to change their after-sales business processes drastically, in a way that is more productive, effective, and less resource-consuming. For instance, prognostic-based knowledge allows to plan a better scheduling of the technical personnel interventions, to reduce machine downtimes, and to decrease overall maintenance costs. Finally, efficient prognostics tools open the way to new business models, such as the shift from selling products to selling services, a practice usually known as servitization.

The availability of cheap Information and Communications Technologies (ICT) enables the application of e-Maintenance to the market of low-cost automated machines, such as those in the class of household and similar appliances (domestic appliances, vending machines, ice cream machines, industrial washing machines, etc.) (Lazzarini et al., 2013). The devices belonging to this class are complex, high-duty systems with long life cycles, usually up to ten years, during which several maintenance services are normally required.

Additionally, a wide range of operating conditions, which might change during machines' lifetime, characterizes them. The maintenance strategy traditionally adopted for such machines is based on on-site maintenance interventions scheduled with a combination of run-to-failure and periodic maintenance policies. However, due to their dissemination on the territory and to the cost of qualified technical personnel, traditional maintenance strategies are very expensive.

The term prognostics typically refers to a systematic approach in CBM whose goal is the prediction of the RUL of a component. Prognoses are based on the continuous monitoring of machine conditions and on several, different types of analysis on the historical operating profile of some device (Jardine, Lin, and Banjevic, 2006).

Prognostics solutions foresee the future performance of a component by predicting its degradation trend and assessing the extent of deviation of the system from the normal operating conditions. Being capable of discovering correlated trends in the system dynamics and of estimating its future performances, prognostics-based maintenance strategies are extremely useful to monitor the health of large scale engineering systems and to enable preventive maintenance (Lazzarini et al., 2014).

Similarly to diagnostics, prognostics-based approaches are composed of three consecutive steps: data acquisition, data processing, and maintenance decision-making (Jardine, Lin, and Banjevic, 2006). Data acquisition is the process of collecting and storing data coming from multiple sources for the purposes of the maintenance. Data acquisition faces several, non-trivial architectural and design choices, such as the what data to record (or what kind of filters to apply to them), the storage support, the protocol stack for communication, etc. The best solution depends on the amount of data a system need to gather and move, on the budget, on the targets of prognostics, and on the characteristics of the devices subjected to maintenance.

Data processing is the process that prepares and then analyzes the data gathered in the previous step. The first, critical phase of this step is data-cleaning, which is necessary to avoid (or, at least, reduce the probability) that data containing errors or inconsistencies are used as input of the following steps. The next phase of data processing has the goal of extracting useful information from the data. (Jardine, Lin, and Banjevic, 2006) and (Johnson, 2011) provide a rich overview on many techniques. Their effectiveness highly depends on the type of data to process.

The decision-making phase takes care of making intelligent, informed, and appropriate decisions about what actions to take, based on the output of the previous phases and on the available resources. The capability of predicting the RUL of components, which differentiate prognostics from other maintenance strategies, has very important consequences in the decision making process. In fact, not only can effective prognostics tools prevent damages to components and avoid costly downtimes, but also can they help to manage the stock of spare parts and the scheduling of the technical personnel.

Current prognostic methods can be classified broadly in model-based and data-driven approaches (Poongodai and Bhuvaneswari, 2013; Dragomir et al., 2009).

Model-based prognostics rely on the existence of a physical model of the component or machine for the estimation of the RUL. Usually, decision-making is applied to the output of the model according to thresholds which are statistically determined.

Uncertainty due to the assumptions and simplifications of the adopted models set limitations on this approach.

The data-driven (DD) prognostics methods use current and historical data to statistically and probabilistically derive decisions and predictions about the remaining lifetime and reliability of an engineering system. The idea behind these methods is that, instead of hardcoding some model, it is possible to automatically build one that fits the historical data.

Data-driven prognostics methods benefit from many interesting properties, such as the ability to transform high-dimensional noisy data into lower-dimensional information (data cleaning/denoising), the ability to discover and handle collinearity in the high-dimensional data, the facilitation of model building through the identification of the relationships underlying the engineering system, etc. (Poongodai, and Bhuvaneswari, 2013). However, DD approaches suffer greatly from scarce and low-quality operational data.

Artificial Intelligence (AI) techniques and/or statistical techniques (e.g., regression methods) are the bases for data-driven prognostic approaches. Data-driven AI techniques comprises conventional numerical methods, such as the linear or gaussian process regressions, and machine learning methods, such as neural networks, decision trees, and support vector machines or relevance vector machines (Schwabacher, and Goebel, 2007; Goebel et al., 2008; Poongodai, and Bhuvaneswari, 2013).

Data-driven statistical approaches achieve prognostics by performing some analysis on the historical records containing the operational conditions and the events of an engineering system. Typically, those data are used to infer the parameters needed to fit a statistical distribution (the Weibull and the lognormal distributions are two examples) that can assess the RUL of the system (Kumar, and Pecht, 2010). Statistical approaches are capable of updating their models, hence of providing better prognostics estimates, as more data become available. Adaptation also make them suitable for systems whose operating conditions might change during their lifetime.

Of a particular interest are the feature extraction-based algorithms, that are typically based on the theory of pattern recognition. These techniques process the operational condition data to extract features that characterize the system and they assume that, in absence of malfunctions, the statistics of those features will remain relatively constant. Current feature values are used to evaluate the health of the system, while their historical trends are utilized to predict the evolution of the damage over time (Pecht, and Kumar, 2008).

The quality of the features chosen to characterize the system is fundamental for the effectiveness of prognostics; different studies present a quantification metric to evaluate and compare their quality based on a separability function (Medjaher et al., 2012).

Similarly, a few studies also expressed the need for metrics that permit to evaluate the effectiveness of prognostics techniques. Prognostics is a relatively new research area in many technological fields, thereby research so far have focused more on finding solutions for its efficacious implementation rather than devising metrics to enable the comparison of different efforts. Some authors reviewed the metrics implemented in several domains that involve prediction and prognostics of some kind, and proposed metrics tailored for the standardized evaluation and comparison of different algorithms (Saxena et al., 2008).

Finally, we would like to point out that prognostics is not a departure from diagnostics. In fact, notwithstanding the fact that prognostics is far more effective than diagnostics in preventing failures and in supporting the intelligent scheduling of technicians, there will always be some faults and damages which are unpredictable. Nonetheless, diagnostics strategies are still effective in timely identifying such events and in taking immediate action to correct the problems.

2.9. Self-Maintenance and Immune Systems

Self-maintenance refers to the ability to carry out regular performance and safety checking by machine itself, to detect anomaly, and to make immediate repairs when needed by using autonomous systems and stocked spare parts to avoid potential catastrophic loss (Lee, Ghaffary, and Elmeligy, 2011).

Self-maintenance is a new design and system methodology. A self-maintaining machine can monitor and diagnose itself, and if any kinds of fault, failure or degradation happen, it can still maintain its basic functions, and inform a remote agent for service support if needed.

Artificial Immune Systems (AIS) are adaptive systems inspired by the biological immune system and applied to problem solving (De Castro, and Timmis, 2002; Timmis et al., 2008). Immune system is a natural defense against pathogens. An adaptive immune system is much more complex than just the immune system itself as it is capable to identify new threats, build a response to them and embody this knowledge. AIS tries to reproduce strategies of an adaptive immune system by acquiring its characteristics (Somayaji et al., 1997).

The idea of Engineering Immune Systems (EIS) is to consider the network as a single body. As in a biological system, if a part of it is attacked, the whole body reacts, e.g., if one or more machines on the network have a problem or are attacked, the others modify their configuration so as to be immune. This requires first the ability to detect and identify anomalies, which is researched in the area of fault detection, diagnosis and isolation. The focus of engineering immune systems is on mechanical engineering systems and on making use of prognostics techniques in order to reach a mechanical system that has self-immunity.

Chapter 3

The Teorema Project

3.1. Introduction

The considerations presented in this thesis emerge from the author's experience in building an innovative e-Maintenance platform for the management of the ice cream machines produced by Carpigiani Group (http://www.carpigiani.com).

3.2. The Scenario and the Need for e-Maintenance

Established in Bologna in 1946, today Carpigiani is the world's top producer of ice cream machines. Carpigiani's ice cream machines are complex and high-duty systems which operate unattended, perform critical food processing operations and have a long (usually more than 10 years) life cycle. As a result, notwithstanding the quality of the construction, during their lifetime ice cream machines require many maintenance interventions, which are performed on-site and are scheduled according to a combination of run-to-failure and planned maintenance policies. Because of the high revenue of ice cream making machines, it is of utmost importance to realize timely detection of failures and deliver prompt technical support interventions. Considered that ice cream machines are installed all over the world, very often in remote geographic areas, on-site interventions for maintenance operations is usually very expensive, for both technical support personnel costs and (possibly long) machine downtimes with related revenue loss. For example, one of Carpigiani's most important customer is in Norway, and his machines are typically installed in remote locations (e.g., Longyearbyen, the largest settlement of Svalbard) where, for most of the year, transport is carried out with boats, helicopters and snowmobiling.

It shall be noted that, in addition to maintenance interventions, ice cream machines need a constant supply of ice-cream base mix, in order to ensure continuous operation. As a result, customers need to place a special attention in supplying their ice cream machines before they are empty.

The advantages that the deployment of an e-Maintenance platform could bring in the management of ice cream machines are evident. e-Maintenance solutions have an enormous, disruptive potential that could completely change the after sales process in the that specific market.

3.3. The Teorema Project Requirements

As seen previously, any e-Maintenance platform in charge of controlling machines of the household and similar appliances class should provide some basic functions to support the principal operations that can be summarized as follows:

a. Automated Monitoring

The automated monitoring function should realize the periodical and/or on-demand (where applicable or appropriate) reporting of the machines' operational state (including failure notifications), collecting maintenance data from the ice cream machines, processing them, and enabling their analysis by technical support personnel through a dedicated management interface. In addition, the e-Maintenance platform should provide advanced reporting mechanisms.

b. Self-Diagnostics and Prognostics

The self-diagnostics and prognostics function should realize both the automated diagnosis at the machine side and the comparative analysis of data collected from all the machines in order to detect failures, anomalous operations, or machines and components that are likely to fail. In case of unexpected behaviour in one ice cream machine, the e-Maintenance platform should immediately notify the specific technician who is in charge of its maintenance.

c. Remote Management

The remote management function should allow to establish on-demand live monitoring and diagnostic sessions to help technicians in quickly identifying the cause of faults on the machines under their responsibility and in deciding how to proceed for the resolution.

In addition, the platform should provide a remote reconfiguration function to enable the automation of firmware and configuration upgrades for the machines.

The remote assistance functions provided by the platform should be specifically designed to allow expert technicians to access and analyze machine maintenance data in real-time from their offices, thus operating as consultants for personnel working on the field. This is essential to optimize the work time of the senior technical service managers.

The set of applications required to implement the above services is characterized by the following functional requirements:

- *Monitoring machine/process status*. Real time gathering and representation of functional parameters including machine and/or process status.
- *Bidirectional communication*. Necessary for managing software updates and changing operation parameters.
- *User authentication*. Control access to the system through a preliminary authentication phase that identifies users and determines their authorizations.
- *Localization*. View networked machines and/or processes.
- *Integration with external computer systems*. Connection with existing ERP and CRM systems.
- *Ability to perform diagnosis and prognosis*. Implementation of advanced statistical and calculation tools of the data gathered.

Similarly, the set of applications required to implement those services will have to be characterized by the following technical non-functional requirements:

- *Availability*. The system must always be available and manage faults without loss of service. Consequently, systems must be designed with an adequate level of redundancy.
- *Security*. The system must guarantee the maximum level of security. The servers must be configured and systematically maintained to avoid possible attacks and prevent data loss. Communication with the server must be made with protected communication protocols over standard ports.
- *Expandability*. The system must be easily expandable and adaptive in case operations in the field raise new needs to be met.
- *Scalability and flexibility*. The system must be scalable and flexible, and it must also be able to meet the growth demands of the network of machines being managed.
- *Multichannel access*. The information gathered on the server must be available through multiple channels so that it can be accessed by any external device (i.e., PC, and so on).
- *Data storage*. Data are stored in a database that becomes the center of all applications developed. Data must be represented in a standard manner (i.e., XML) and size (number of events gathered per machine/process, number of events available online, and so on).
- *Data separated from the service*. There must be a distinction between data from the machines, their presentation (business logic, authentication, etc), and service logic (analysis, reports, data mining).
- *Management of data traffic*. Sending of multiple data sets to the server using the same connection (and therefore at the same cost), like client IP, GSM cell info for localization, GSM timestamp.
- *Code protection*. Possibility of hiding and making source code inaccessible from the outside.

For programming and code development it was decided that scripting languages are best suited for developing e-Maintenance applications in the Industrial Internet of Things (IoT). The use of interpreted languages provides great flexibility, rapid development (without the need for compilation it is possible to update the code and restart without having to rebuild the source), and code portability. It was also evaluated using innovative Open Source technologies that make it possible to rapidly develop systems (RAD – Rapid Application Development) and agile methods, keeping in mind two concepts: Don't Repeat Yourself (each piece of information in the system should be expressed in a single location) and Convention over Configuration (there are default solutions and conventions for almost all aspects of applications).

3.4. The Teorema Platform

Teorema is the Carpigiani integrated platform for e-Maintenance. Teorema platform is based on a client-server approach and enables a variety of techniques and services, the main of which are described in general below.

3.4.1. Teorema Operations

Teorema provides several services, which include automatic notification of malfunctions; synoptic visualization of maintenance data; automated reporting for monitoring and diagnostic purposes; prognostic functions to identify machines and components that are likely to fail; and remote assistance interventions, such as interactive diagnostic analysis, reconfigurations, and firmware updates. The functions provided by Teorema as well as the technologies adopted for their implementation are proving robust and effective. Teorema information and services can be exploited using different ICT media.

Figure 2, 3, 4, and 5 depict examples of screenshots and typical data provided by Teorema.

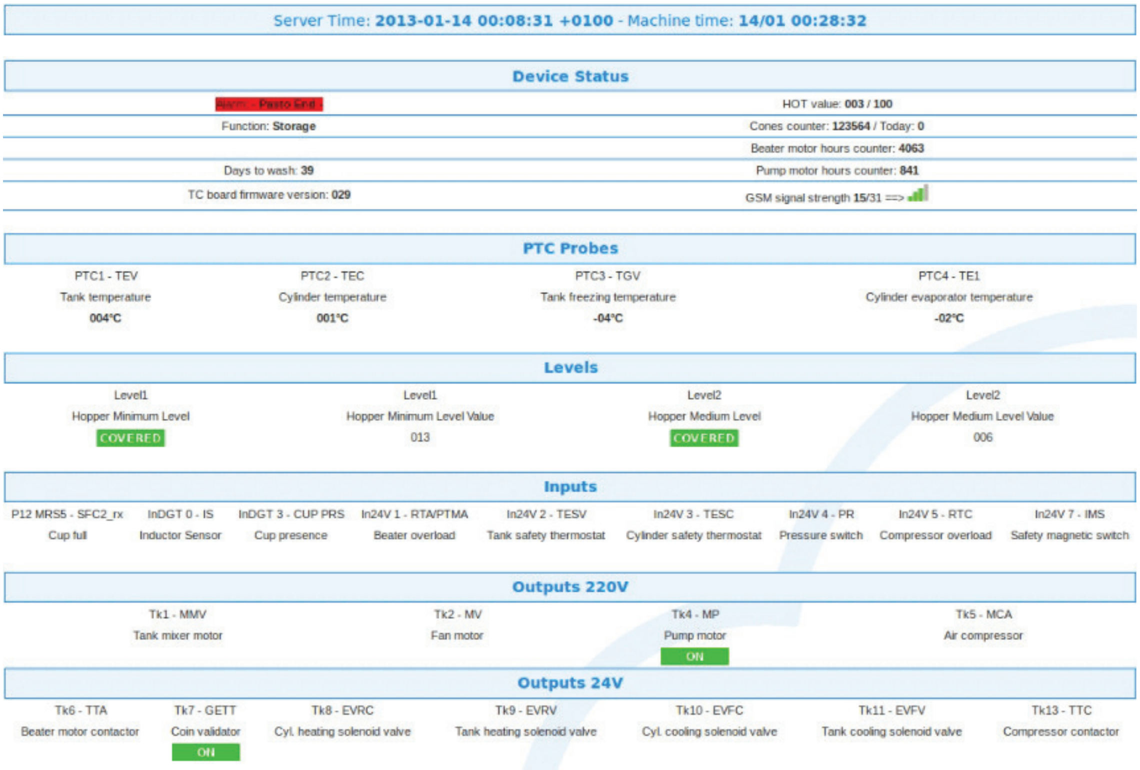
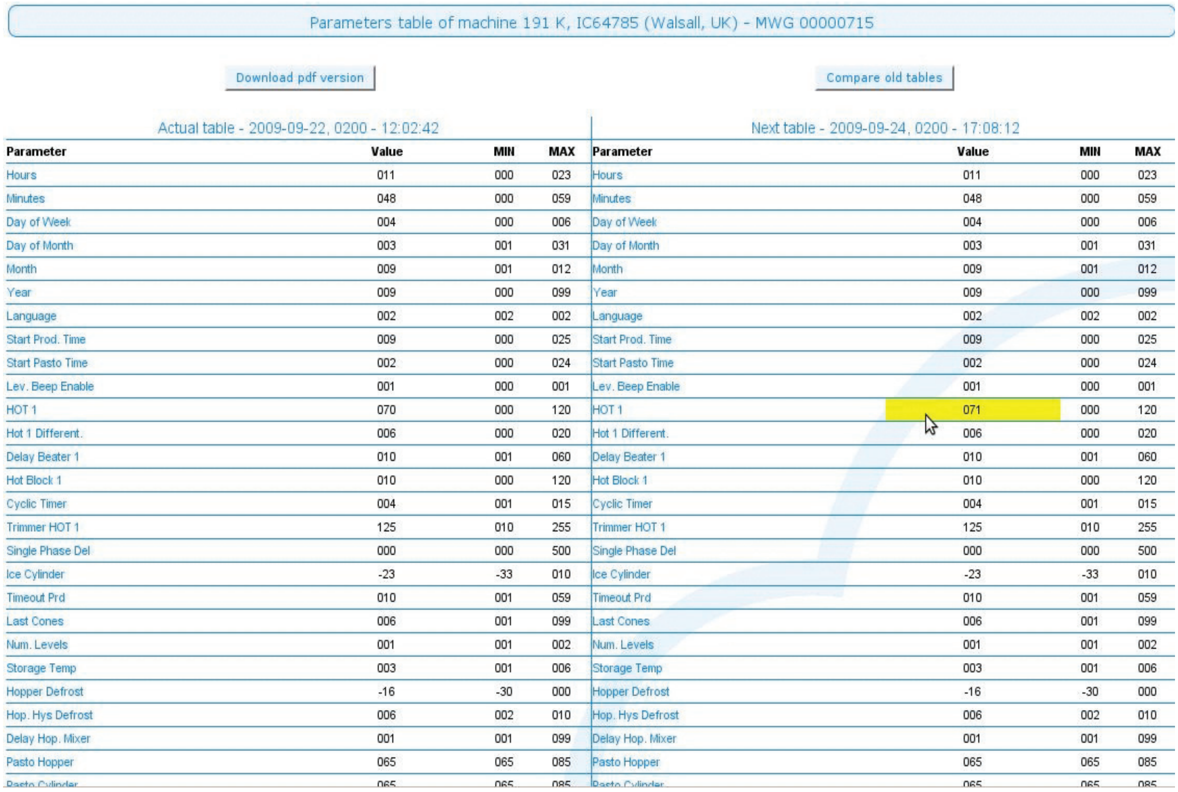


Figure 2 – Web application – Machine Status

Figure 3 – Web application – HACCP Data



Parameters table of machine 191 K, IC64785 (Walsall, UK) - MWG 00000715

Download pdf version | Compare old tables

Actual table - 2009-09-22, 0200 - 12:02:42

Parameter	Value	MIN	MAX
Hours	011	000	023
Minutes	048	000	059
Day of Week	004	000	006
Day of Month	003	001	031
Month	009	001	012
Year	009	000	099
Language	002	002	002
Start Prod. Time	009	000	025
Start Pasto Time	002	000	024
Lev. Beep Enable	001	000	001
HOT 1	070	000	120
Hot 1 Different.	006	000	020
Delay Beater 1	010	001	060
Hot Block 1	010	000	120
Cyclic Timer	004	001	015
Trimmer HOT 1	125	010	255
Single Phase Del	000	000	500
Ice Cylinder	-23	-33	010
Timeout Prd	010	001	059
Last Cones	006	001	099
Num. Levels	001	001	002
Storage Temp	003	001	006
Hopper Defrost	-16	-30	000
Hop. Hys Defrost	006	002	010
Delay Hop. Mixer	001	001	099
Pasto Hopper	065	065	085
Pasto Cylinder	065	065	085

Next table - 2009-09-24, 0200 - 17:08:12

Parameter	Value	MIN	MAX
Hours	011	000	023
Minutes	048	000	059
Day of Week	004	000	006
Day of Month	003	001	031
Month	009	001	012
Year	009	000	099
Language	002	002	002
Start Prod. Time	009	000	025
Start Pasto Time	002	000	024
Lev. Beep Enable	001	000	001
HOT 1	071	000	120
Hot 1 Different.	006	000	020
Delay Beater 1	010	001	060
Hot Block 1	010	000	120
Cyclic Timer	004	001	015
Trimmer HOT 1	125	010	255
Single Phase Del	000	000	500
Ice Cylinder	-23	-33	010
Timeout Prd	010	001	059
Last Cones	006	001	099
Num. Levels	001	001	002
Storage Temp	003	001	006
Hopper Defrost	-16	-30	000
Hop. Hys Defrost	006	002	010
Delay Hop. Mixer	001	001	099
Pasto Hopper	065	065	085
Pasto Cylinder	065	065	085

Figure 4 – Web application – Parameters Setting

Figure 5 – Mobile App

3.4.2. Remote Management

The remote assistance functions provided by Teorema have also allowed the performance of several remote maintenance operations. For instance, recently a UK-based customer requested the customization of about 2000 already installed Carpigiani ice cream machines. Using Teorema, the Carpigiani technical personnel managed to accomplish this task by simply upgrading the machine firmware online. The Teorema remote reconfiguration functions have also enabled the adoption of new, significantly facilitated machine installation and configuration procedures. In fact, ice cream machines now do not need to be preconfigured in the factory or on-site but are simply installed and configured online. In addition, in the last few months, Carpigiani has performed hundreds of remote reconfigurations of Teorema-controlled machines to tune their operating parameters, for example, scheduling of the pasteurization process, according to the specific deployment location.

3.4.3. Diagnostics

The Teorema diagnostic functions enable Carpigiani technicians to perform many assistance operations remotely. More specifically, Teorema allows technical support personnel to trigger real-time on-demand monitoring sessions on a specific ice cream machine, thus enabling them to retrieve the desired maintenance information and perform remote assistance interventions via a handy and ubiquitously accessible Web interface. In case an on-site intervention is necessary, technicians can leverage on Teorema-provided information to identify and carry with them the required spare parts, thus reducing the need for further on-site operations. In addition, real-time access to diagnostic information allows users to make sure that ice cream machines do not experience costly downtimes due to improper or untimely management by operating personnel, for example, unjustified delays in the loading of ingredients or in the cleaning of the machine. For instance, in a recent maintenance intervention, a Carpigiani technician received an alarm from a Teorema-enabled machine and managed to trace the source of a problem to an obstruction in the ventilation grid—a misuse that could have led to serious damage to the machine. The technician immediately contacted the customer, who confirmed that there was a folded cardboard box leaning on the back of the machine and removed it, thereby fixing the problem.

3.4.4. Prognostics

The Teorema prognostic functions have also proven capable of detecting anomalous situations and of identifying which components are likely to break in underperforming machines, especially those involved in the cooling process. For instance, in one machine, Teorema detected the upcoming failure of the beater blades component by identifying a negative trend in the ratio between the number of served ice cream cones and the accumulated run time of refrigeration compressors.

Teorema prognostic information has enabled Carpigiani to adopt a proactive maintenance strategy, which allows the optimal scheduling of on-site assistance interventions according to the machines' current operating conditions with significant reductions of machine downtimes. The remote detection of the type of assistance intervention required for failed components allows the optimization of (in terms of specialized personnel and spare parts) on-site assistance interventions, which now are better planned and scheduled.

In addition, Teorema has enabled a more effective use of work time from expert technicians, a critically scarce resource, who now can be much more productive as they do not have to spend most of their time traveling to gain on-site access to the ice cream machines. The ubiquitous Web-based access to all maintenance data also facilitates the cooperation of technical support personnel operating in the field with the experts of the Carpigiani management division working at the Carpigiani Headquarters.

However, despite these very good results, Teorema has the potential to further increase savings. With the introduction of prognostics techniques, in fact, Teorema is paving the way to a drastic change in the after-sales processes of Carpigiani.

The prognostics-based tools of Teorema aims to estimate the RUL of the most critical components of ice cream making machines, in order to prevent failures and costly machine downtimes by early recognizing abnormal behaviors, glitches, and other minor issues that would eventually grow into more serious problems. In addition, the increased knowledge on the RUL of components represents an extremely useful feedback for designers at Carpigiani, who are thus able to focus their work on improving the most critical components.

The prognostics techniques implemented analyze the data coming from all the Teorema-based ice cream making machines. The automated monitoring and the remote management functionalities of Teorema take care of data collection and of their storing in the servers at the Carpigiani Headquarters. The previous step is necessary to have enough computational power to process those data, applying the necessary aggregations and manipulations, and then to use them to perform an in-depth statistical analysis. This analysis permits to derive a new information, like the discovery of recurrent patterns in some specific metrics that usually precede the occurrence of problems, which is more valuable than the simple record of machines' operational states.

The amount of data received from the machines and stored in the servers is huge: to date, the database contains the entries collected since the beginning for all Teorema-based machines. The data format is that of raw operational state snapshots, and so they need to undergo the preprocessing phase before they can be used for statistical analyses. Moreover, while each snapshot contains the value of many different parameters (temperature and pressure, various timers, total and daily number of cones produced, alarms generated, current machine time, etc.), there are some important metrics which are not explicitly available.

For instance, there are no values in the snapshots generated by the ice cream making machines that represent the duration of the pasteurization cycle. However, it is possible to process the information stored in multiple snapshots in order to derive that value. When the cycle begins, the pasteurization machine rises the temperature of the mixtures contained in the tank and in the barrel to 65°C. The longest time between the two determines the duration of the heating phase. At this point, the machine keeps the temperature of the mixture in both the containers at 65°C for 30 minutes, when the cooling phase begins. The duration of this phase is also variable, and it is determined by the time necessary to bring the temperature of the ice cream mixture in the tank and in the barrel down to 4°C.

The four points that identify the curve, i.e., beginning of the pasteurization cycle, end of the heating phase, beginning of the cooling phase, and end of the pasteurization cycle, are associated to different alarms which are all codified into the snapshots. Therefore, it is possible to extract the exact sequence of alarms from the snapshots and to recreate the pasteurization cycle from it. It is important to note that the removal of any incomplete cycle (data cleaning phase) is essential in order to analyze consistent data. The statistical analysis performed on the historical trend of the duration of the pasteurization cycle revealed the importance of this metric for the purposes of maintenance.

3.4.5. Customer Access to Production Data

Teorema enables Carpigiani to provide an improved and innovative service to its customers. In fact, Teorema allows Carpigiani to provide direct access to both production and maintenance data to its customers.

Providing customers with real-time access to production data, such as machines productivity metrics (number of draw operations, cycle time, daily ice cream dispensed by flavor, etc.), working parameters, reliability indicators, and geographic location, represents a significant innovation in the ice cream machines market. In fact, the availability of real-time temperature data during the refrigeration process is a very important feedback that significantly facilitates the development and preparation of peculiar ice cream recipes.

This feature has been much appreciated by customers, which in some cases have asked Carpigiani to tune their ice cream machines working parameters to ease the production of their signature recipes. In addition, customers can now leverage on Teorema-provided temperature data related to pasteurization and refrigeration processes to demonstrate their compliance to food safety rules to the control and certification authorities in their country.

Finally, customers take advantage of Teorema to access information about the machines' assistance history, including up-to-date reports on the status of ongoing maintenance interventions.

3.5. The Teorema Architecture

Teorema adopts a client/server architecture, with machine-installed kits that integrate with the ice cream machine hardware and act as clients sending maintenance data to the Teorema server, a centralized data monitoring and control station (currently located in the Carpigiani Headquarters).

Internet-based communications allow to take advantage of a pre-existing communication infrastructure and a wide range of network connectivity media, e.g., 802.11, GPRS, and Ethernet, which is essential to enable the deployment of Teorema-enabled machines in a wide range of deployment scenarios. In addition, open standards and the widespread availability of COTS hardware and software components significantly help to reduce costs. Different Teorema operations are executed at either the client or the server side, with a carefully designed placement that is decided considering the requirements in terms of computational capability and communication costs. For instance, the diagnostics operation is distributed across both client and server components, while the monitoring follows a centralized approach.

Figure 6 depicts the interactions between the components of the Teorema platform.

Diagnostic functions are split between the ice cream machines and the central monitoring station, because the limited CPU power available on the machines requires to run locally only simple self-diagnostic functions (based on either predefined threshold values for monitored variables or alarm triggered by machine-installed sensors) and to transmit maintenance data to the central monitoring station for aggregation and further processing. On its turn, the Teorema central station performs just-in-time analysis of the received data via computationally expensive diagnostics and prognostics algorithms, further increasing the capabilities to timely detect failures and unexpected behaviours.

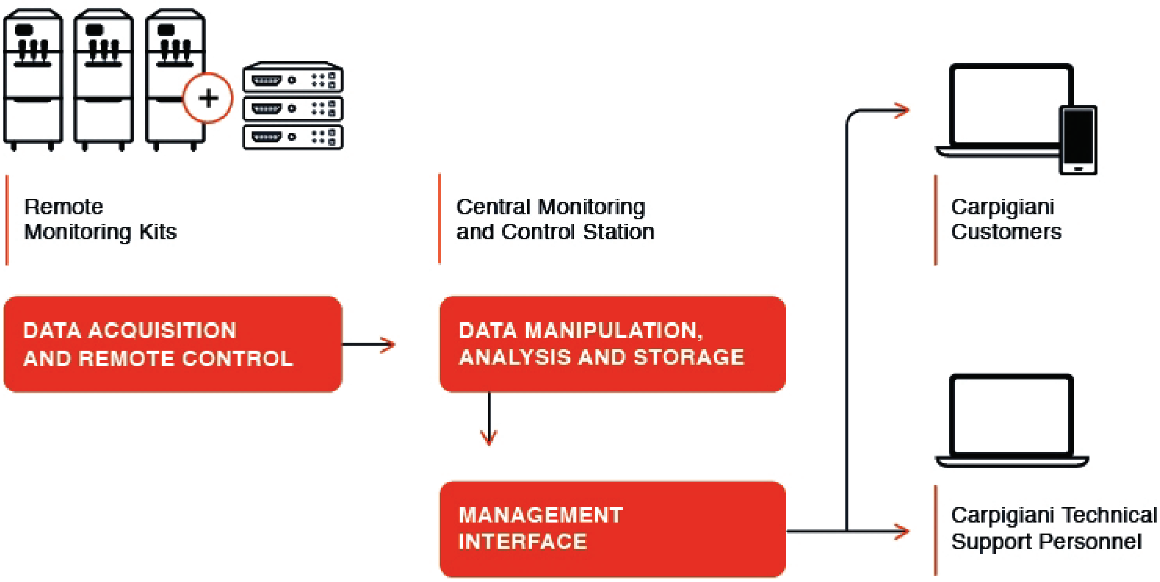


Figure 6 – The Main Components of the Teorema Platform and their Interactions

The monitoring of machines is, instead, fully centralized. In fact, only the central management station has the storage capacity to record all the management data and the computational power to process them to provide aggregation and visualization functions. The transmission of maintenance data from the ice cream machines to the central station occurs according to several conditions. The machine-installed diagnostics functions automatically report to the central station in case of malfunctions. In addition, periodical transmissions of maintenance data are also scheduled. Finally, the machines also provide a control interface for remote technical support interventions, enabling on-demand, real-time transmission of monitoring data.

Most of the data received by the central station consist of discrete data sets, also known as "value type" monitoring data in maintenance literature (Jardine, Lin, and Banjevic, 2006), which are more difficult to handle than continuous time series data and present some limits to the applicability of automated anomaly detection mechanisms for real-time monitoring systems (Chandola, Banerjee, and Kumar, 2009). In addition, ice cream machines also record a list of events, including events related to the normal operating conditions of the machines such as commands issued by the machine operators, completion of processes such as pasteurization, etc., and transfer them to the central management station. This information are the so-called "event data" which we have discussed previously

The centralized monitoring and control station, called Teorema server, represents the core of the e-Maintenance platform, as it collects, stores, analyzes, and provides reports on maintenance data. Web clients, running on top of the Carpigiani WebGate (CWG) machine-installed telemetry kit, transfer maintenance data to the Teorema server and enable the remote control of Carpigiani's ice cream machines. Carpigiani WebGates communicate with the Teorema server through the Internet, using a TCP/IP stack, leveraging on one of the supported connectivity media to access the network.

3.5.1. The Teorema Server

The Teorema server is made out of 5 components, i.e., the Data Collection and Manipulation (DCM), the Presentation Interface (PI), the Monitoring Controller (MC), the Diagnostics and Prognostics Module (DPM), and the Reporting Module (RM).

Figure 7 depicts the interactions between the components of the Teorema server.

PI is a Web application which allows the synoptic visualization of ice cream machine operational state. PI has several access levels, e.g., technical support and customer, and provides the user with a subset of its functions according to his role. PI is also multimodal, therefore being accessible anytime anywhere from mobile devices, and leverages on mashup services, e.g., Google Maps, to provide information about machine context (e.g., location).

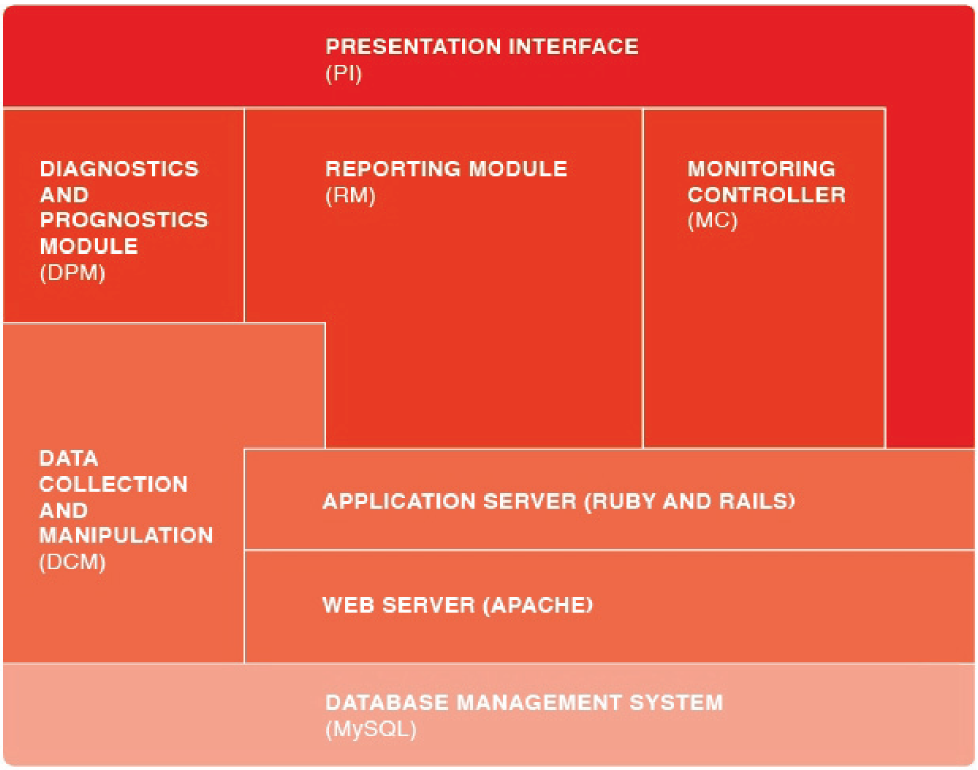


Figure 7 – The Teorema Central Monitoring and Control Station Architecture

DCM collects the maintenance data received from CWG, normalizes it, registers the collected information in a DBMS, and notifies CWG about the availability of firmware and/or configuration upgrades.

MC allows technicians to trigger the setup of a real-time monitoring session on a particular machine. Monitoring sessions permit to retrieve specific maintenance information and perform remote assistance interventions via a handy and ubiquitously accessible Web interface. MC also implements authorization and logging functions for remote assistance operations.

RM takes care of aggregating monitoring data stored in the database, produces a daily report for each ice cream machine, and sends the report via email to the technical support personnel in charge for the specific machine.

Finally, DPM analyzes collected data to detect anomalies in the machine state in order to identify malfunctions and unexpected conditions such as underperforming cooling process or components which are likely to fail. The just-in-time analysis of operational data performed by the Teorema central monitoring station performs both the detection and the prediction of failures and therefore allows to schedule on-site assistance interventions accordingly.

3.5.2. The Carpigiani WebGates

Carpigiani WebGates enable the monitoring of machine operational state such as the number of ice creams served, the hours of work performed by the beater motors, the temperatures reached in critical components of the machine, the viscosity of the served ice cream, the results of the pasteurization process, the air pressure in the ventilation grid, etc. Carpigiani WebGates communicate with the Teorema server using a push-pull paradigm. They automatically issue a urgent report to the central monitoring and control station in case of malfunctions.

In addition, they periodically transfer maintenance data, as well as the events occurred during machine operations, to the Teorema server. Finally, Carpigiani WebGates support the continuous real-time transmission of monitoring data to the central station, enabling on-demand remote diagnostic sessions.

The CWG is realized as an extension board which interfaces with Carpigiani's proprietary diagnostics interface to collect operational data, check failure reports from machine-installed sensors, change configuration parameters, and update firmware and software.

Carpigiani WebGates continuously monitor the current machine state and report it to the Teorema server. They also provide a remote control interface that allows technical support personnel to connect to the ice cream machine, retrieve information in real-time, and perform remote assistance interventions. In addition, the CWG can automatically download and upgrade both the machine and the extension board set of configuration parameters and firmware.

Finally, in case of anomalous machine behaviour, Carpigiani WebGates immediately initiate a monitoring session that reports an alarm to the central monitoring station. In case of major faults, the diagnostics board can also decide autonomously whether to stop the ice cream machine.

Carpigiani WebGates have a network interface which allows them to communicate with the Teorema server. Given the wide range of installations of the Carpigiani ice cream making machines, we have designed four types of WebGates, with different types of Internet access technologies: IEEE 802.3 (Ethernet), IEEE 802.11 (Wi-Fi), Echelon Corporation LonWorks, GPRS and UMTS network. Currently, the most deployed connectivity media is GPRS, as it represents a good compromise between traffic costs and ubiquitous access and it permits to access machines without the need to modify customers' security policies, e.g., network routing and firewall configurations. The GPRS solution is based on Internet protocols, allowing the realization of applications and services with data transfer rate up to 40 kbit/s and low communication cost.

The CWG can be installed on every ice cream machine model of the current Carpigiani production and can be installed also in many older models already in operation.

Figure 8 depicts an example of the CWG installed on an ice cream machine.

Figure 8 – The CWG Remote Monitoring Kit Installed on an Ice Cream Machine

3.5.3. The Teorema Database

The Teorema platform uses a database as a center of all applications which collects data from disparate sources. This database receives and stores data from the machines, and users and is developed in a way that supports all the processes for the remote assistance service.

The core of the database is the table *machines* where the information of the equipment are stored. The table machines is linked to the other tables from which obtained part numbers, firmwares, session information (i.e., monitor or alarm), and so on.

Each machine can be part of one or more groups of machines. Each group of machines can belong to one or more user groups. Each group of users can be associated with one or more users. All users registered in the system are located in the table *users* that contains all the related data (username, password, email, etc.) and makes them available to the system. The database design for the Teorema platform is given in Appendix A.

3.5.4. The Teorema Communication Protocol

Teorema is based on a client/server architecture, where the roles of client and server are held by the WebGate and the Teorema Daemon.

The daemon provides the service of collecting data, receiving and storing operating data and the status of the machines, and making it available, using the Internet, to the clients.

This application is considered the centre of the infrastructure because it provides the service described above, but also deals with the communication management of the system through the implementation of access techniques, resource allocation, data security, and so on. The code of the Teorema Daemon is given in Appendix C.

The communication protocol between the Carpigiani WebGates and the Teorema remote server has been developed ad hoc for this specific application. The protocol refers to the Internet protocol stack TCP/IP, but a native proprietary choice was made. The Teorema Protocol is based on the interchange of simple strings of ASCII characters (DOS-like). All strings end with the character represented by the ASCII code CR (Carriage Return) and the first characters identify both whoever carries out the request and the type of request made (e.g., alarm, monitor, etc.). The size of the data packet (e.g., alarm snapshot) is 173 bytes plus 4 bytes of header.

The main reason to communicate through a simple strings-based protocol is dictated by the fact that the data transmission involves a cost.

The information exchanged between the parties are of different nature and these include snapshots, programming tables of operating parameters, requests for identification of the type of electronic board connected to the WebGate. In addition, information to verify data integrity (e.g., CRC checksum, MD5 hash) are also transmitted, along with notifications of successful or unsuccessful reception.

Figure 9 depicts the relations and the sub sequent data transmission between the Teorema Daemon and the Carpigiani WebGates.

Assuming that the CWG has already registered the SIM on the GSM/GPRS network and has completed synchronization with the electronic board of the machine, the connection may be triggered by two events, i.e., an alarm from the machine or a call from the server. The connection set up is initiated with the daemon listening for connection requests on dedicated TCP port. When the CWG sends a request, a socket is created. After the connection is established, a fork call creates a child process. Before starting the data exchange, it is necessary to properly perform authentication process.

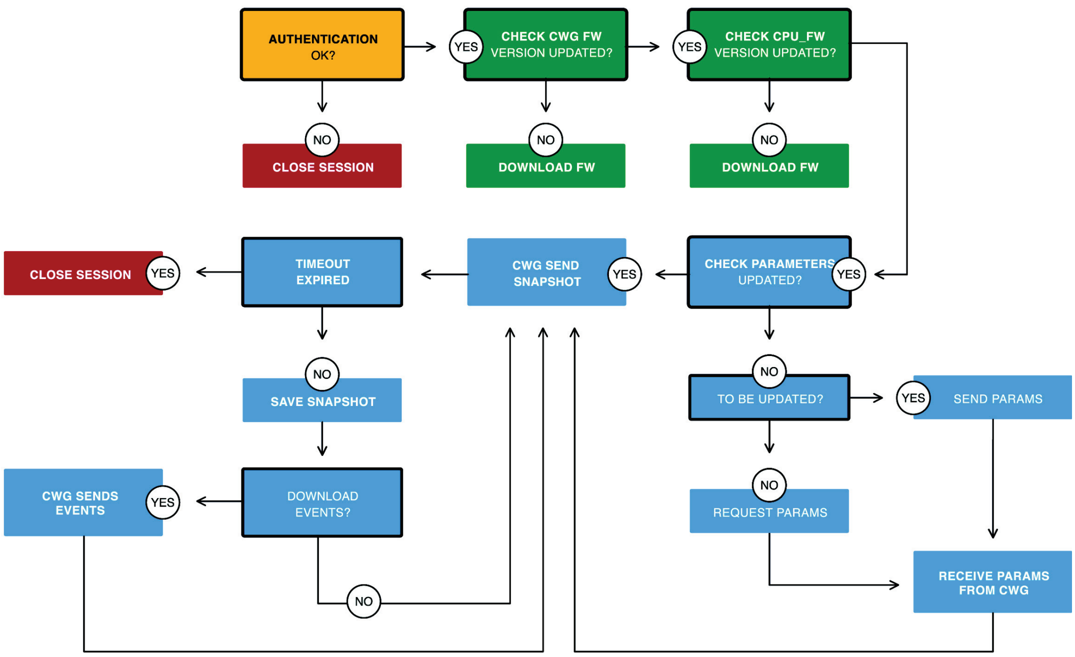


Figure 9 – The Schematic of Teorema Communication Protocol

The daemon has the choice to accept or to reject the connection set up request. If the responder accepts the connection it proceeds with the check for software updates. Then, the demon performs the planned tasks. In case of rejection, the demon closes the connection.

The state diagram of the Teorema Protocol is given in Appendix B.

3.6. Implementation Insights

The development of the Teorema platform is exclusively based on open source technologies, as they allow Carpigiani to protect its investments by not restricting the software platform to a single proprietary vendor. In particular, the technologies adopted are highly innovative, albeit already mature for the industrial use and very well supported.

The Presentation Interface is realized as a Ruby on Rails application. Ruby on Rails is a framework for the development of Web 2.0 applications based on the Ruby programming language and purposely designed to enable agile programming practices (Vosloo and Kourie, 2008). The Ruby on Rails characteristics allow the rapid development of new features, minimizing their time-to-deployment (Bachle, and Kirchberg, 2007). This makes Ruby on Rails a perfectly suited framework for a dynamic and rapidly-evolving platform such as Teorema.

Ruby on Rails is based on Model-View-Controller (MVC) architecture.

The *Model* is responsible for maintaining application state, which can be transitory (duration equivalent to the time required for a few user interactions) or permanent (saved in a database). A model represents not only the data but also defines the business rules that must be applied to that data. Including the implementation of rules in the model assures that no other application element can invalidate the data. In other words, the model simultaneously functions as guardian and container for the data.

The *View* generates the user interface, in general, based on model data. A view can present the user with different data input methods but never manages their entry. The viewing activity terminates as soon as the data are displayed. There can be different views that access the data of the same model, in general, for different purposes but for each action (i.e., method) a view is necessary.

Controllers coordinate the application, receiving events from the outside world (in general by means of user input), interact with the model, and supply users with appropriate views. Figure 10 depicts how the three components work together, i.e., the browser send the request (1), the controller interacts with the model (2), the controller invokes the view (3), the view renders the next browser screen (4).

Rails imposes a workspace in which models, views, and controllers are separated into functional fragments that are connected only when the program executes.

The compilation process is based on predefined *intelligent* values so it is not necessary to provide external configuration metadata to make the application work (Ruby, Thomas, and Heinemeier-Hansson, 2013).

In addition to being a *full stack* framework, Rails can be regarded as a domain-specific language (DSL).

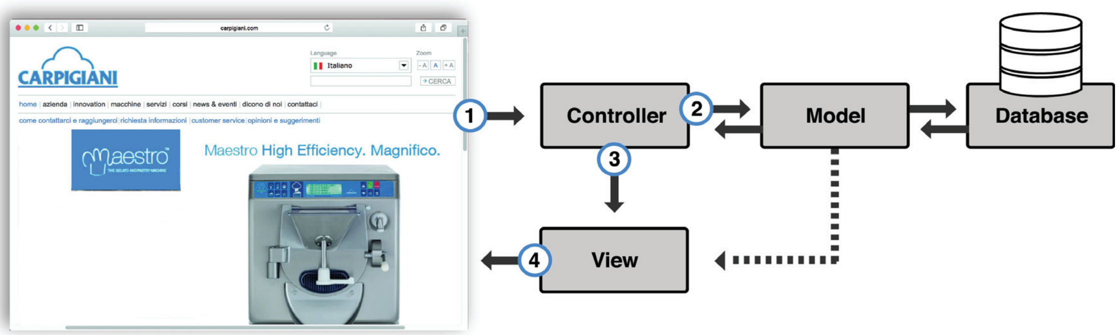


Figure 10 – The Model-View-Controller Architecture

Portions of the Data Collector and Manipulation component are written in Erlang to guarantee extremely high scalability and reliability levels. Erlang is a programming language for highly concurrent and fault-tolerant distributed applications (Vinoski, 2007a) (Vinoski, 2007b). The excellent Erlang support for highly scalable server application is proved by its adoption as the technology to power massively parallel applications such as the Yaws Web server, the ejabberd Jabber server, and the Facebook chat infrastructure (Letuchy, 2008). The Erlang support for hot code swapping also allows to update applications without shutting them down and restarting them, thereby significantly reducing the system downtime.

The data collected from the remote machines is stored in a database operated by MySQL, a relational DBMS well known for its excellent performance and reliability (Di Giacomo, 2005). The 5.1 release of the MySQL DBMS adopted in the Teorema project introduces some enterprise class features, such as table partitioning, that enable to achieve very good performances even in case of very large data sets, thereby suiting the needs of an ambitious and evolving project such as Teorema. Last but not least, the availability of high performance drivers for the Ruby and Erlang programming languages was a major reason for the adoption of MySQL in the Teorema project.

The Teorema Daemon in written in Perl using the Perl Database Interface (DBI) module. The DBI defines a set of methods, variables and conventions that provide a consistent database interface independent of the actual database being used. The Teorema Daemon is using DBI to interface with database for saving and exchanging data.

The Carpigiani WebGate kit is currently realized as a custom board based on a hardware microprocessor platform for embedded applications. Future developments of the WebGate will be based on higher performance hardware platforms with ARM processor, that will allow full fledged operative system while preserving important parts of the current software code design. Some recent COTS boards based on ARM have Linux distributions.

If not, it’s possible to build a custom distribution, starting from a base core and adding the needed packages and libraries. This can be done by using Yocto (http://www.yoctoproject.org), an open source collaboration project that provides templates, tools and methods to help you create custom Linux-based systems for embedded products regardless of the hardware architecture.

During the development of the e-Maintenance platform, a special care was dedicated to the security aspects. Teorema adopts a role-based authentication system and of a set of thorough data sanitization policies, and leverages on encrypted communications for sensitive information. The Ruby on Rails framework was also chosen for its resilience to the most common security attacks for Web applications, such as SQL injection, cross-site scripting, and cross-site request forgery (Poweski, and Raphael, 2010).

3.7. Innovative Sensors

Innovative sensors and techniques are essential to enable different diagnostic, prognostic and e-Maintenance activities. In the present context, three new developments in technological tools for rapid and in-situ control and monitoring, using wireless biosensor network, are implemented.

Firstly, Carpigiani has identified the opportunity to develop a portable biosensor system for bacterial concentration detection based on the impedance technique, that is fully automated and requires no particular knowledge of microbiological techniques, thus making it particularly suitable for microbial screening in industrial or commercial environments (Cocchi, and Lazzarini, 2010a; Cocchi, and Lazzarini, 2010b; Grossi et al., 2010; Grossi et al., 2013). Moreover, the use of cheap electronics makes the new solution highly competitive in terms of cost and this helps to promote a widespread use of it.

In addition, a novel technique to control ice cream freezing by electrical characteristics analysis, tightly linked to temperature and viscosity, provides a suitable, nondestructive tool to monitor ice cream quality product that overcomes the drawbacks of the standard methods (Cocchi, and Lazzarini, 2011, Grossi et al., 2011).

Finally, electrical characterization of products is gaining increasing interest in the food industry for quality monitoring and control. In particular, in the case of ice cream industry, where machines dedicated to store ice cream mixes are programmed *ad hoc* for different groups of products. To this purpose, an automatic ice cream characterization by impedance measurements for optimal machine setting is implemented (Lazzarini, and Cocchi, 2011; Grossi et al., 2012).

The most important aspect of the system is that it is provide an immediate local information but the analysed data will be transmitted to the Teorema platform. Figure 11 depicts a schematic representation of the sensor system.

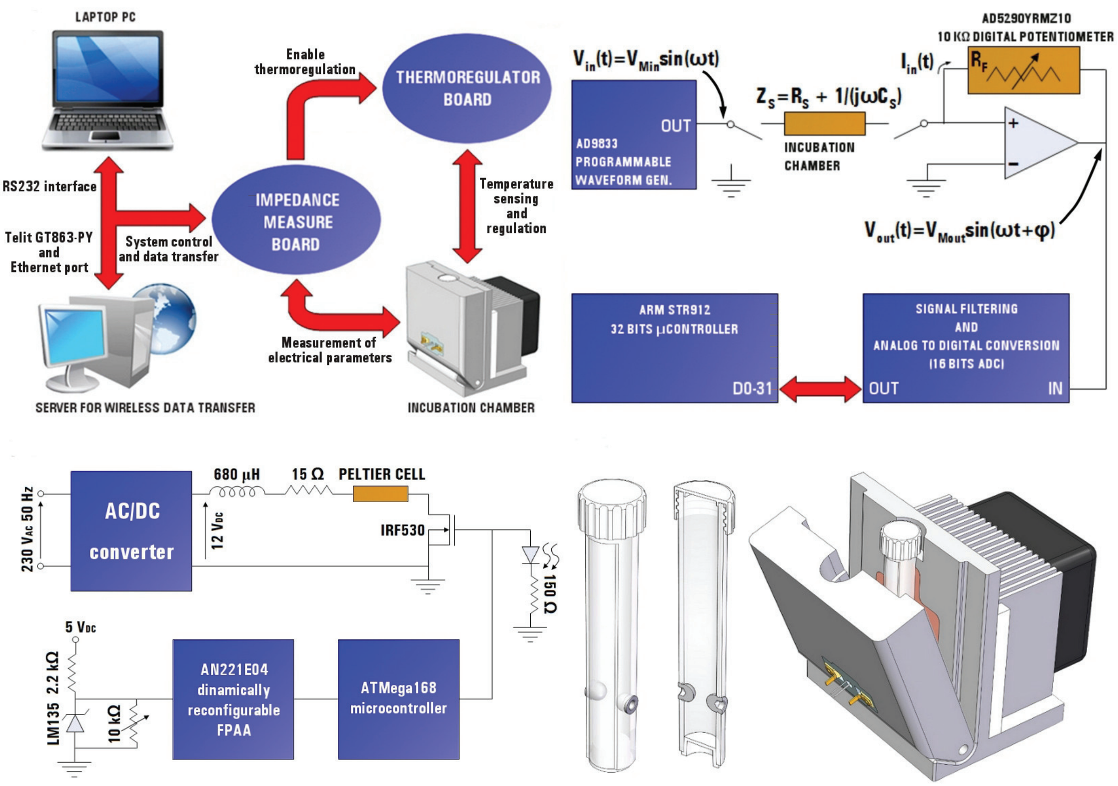


Figure 11 – The Schematic of the Sensor System

3.8. New Product Development

Data collected from the field by Teorema are used to create more intelligent industrial products. Data-driven methodology provides the guidelines for designing and evolving projects based on collected data about how current products are performing in the market. This processes brings Carpigiani to consider the concepts of open innovation, user innovation and data-driven continuous design. Continuous design has two main principles; the first principle requires you to abandon the concept of final designs, i.e., engineers must be involved throughout the entire product development process and not just at the beginning; the second principle requires to realize versatile prototypes, instead of document-based deliverables. The design of new product is becoming so similar to the rolling release in software development. Rolling software, in fact, is continually updated, in contrast to standard release software which is upgraded between versions.

3.9. Servitization

In this context, the term "servitization" refers to the new business models enabled by Teorema. Teorema prognostics information fosters a drastic shift in the maintenance strategy of Carpigiani, which has moved from a reactive strategy to one based on prediction and proactivity. Nonetheless, the benefits of prognostics go well beyond reducing maintenance costs and providing feedbacks to the R&D division. In fact, prognostics techniques produce new, invaluable knowledge that can support many phases of the machine management process and that can profoundly change the after-sales business model. Traditionally, due to the worldwide distribution of their machines, Carpigiani outsources the after-sales maintenance of machines to specialized local companies. The introduction of the Teorema e-Maintenance permitted to reduce the costs associated with the outsourcing of the maintenance, as it enables Carpigiani's technical personnel to perform many assistance operations remotely.

In some cases, in fact, the remote reconfiguration of a few machine parameters is enough to fix the problems. Moreover, if the on-site intervention is necessary, Teorema can help technicians in identifying the required spare parts, so that they can carry them on-site since the first intervention and avoid further travels (i.e., first-time-fix). In addition, prognostics functions enable the early detection of components which are likely to break in underperforming machines, thus giving the technicians the possibility to schedule on-site interventions before breakdowns occur. This is of critical importance, as expert technical personnel is an expensive and scarce resource, and bringing forward problems detection is paramount to schedule their use more efficiently.

Thus, Teorema platform has permitted Carpigiani to reorganize its range of services adding to the standard offer of manpower, parts and extended warranty programs four new packages:

a. Care Plus

Care Plus is an optimal maintenance program to prevent breakages and downtime. Care Plus knows the workflow of the machine and anticipates servicing to lengthen its lifetime. The system monitors the life of the machine based on actual use (not time) and schedules the substitution of critical parts only when necessary.

b. Safety Plus

Safety Plus ensures constant attention to all aspects of food safety and hygiene. It controls each phase of the work cycle (i.e., pasteurization, production, conservation, cleaning) ensuring all standards for HACCP are met. It also provides detailed reports that document the compliance and supplies all the information necessary to answer any enquiries during a health inspection.

c. Efficiency Plus

Efficiency Plus helps to organize the operative management of the cleaning and refill cycles, based on the consumption and actual production volume, to avoid waste.

The system identifies the operative workflow and cleaning days, indicates the correct amount of mix to add according to the wash cycle chosen, and so helps to save mix and avoid washing the machine due to incomplete pasteurization.

d. Energy Plus

Energy Plus analyzes all the details of the production needs and can advise how to save energy. It will always recommend the most efficient use of the machine, based on the actual business needs. The system indicates if the machine is ready to produce or has to be switched to conservation according to the specific needs of the customers and communicates daily average consumption.

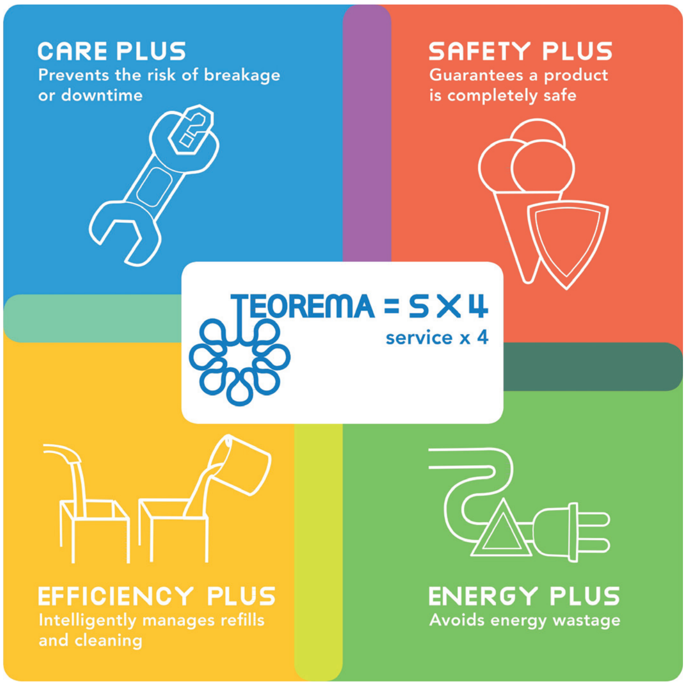


Figure 12 – The Commercial Presentation of Teorema Packages

Figure 12 depicts a schematic presentation of the Carpigiani service proposal.

A mobile app provides all the machine's real-time data for all packages to the customers.

Finally, Carpigiani is now considering the possibility of insourcing the after-sales maintenance of machines. The deep knowledge of each machine operativity is also pushing Carpigiani to consider servitization, i.e., moving from products to services. In the case of Carpigiani, this would mean moving from a business model centered on selling ice cream making machines to a new one, characterized by the selling of "a number of ice cream cones" which will be served in a store location.

3.10. Summary and Concluding Remarks

Project Teorema is already in production with 5550 remotely controlled machines installed in more than 20 countries around the world. Figure 13 depicts the coverage of the Teorema platform and figure 14 the installed base.

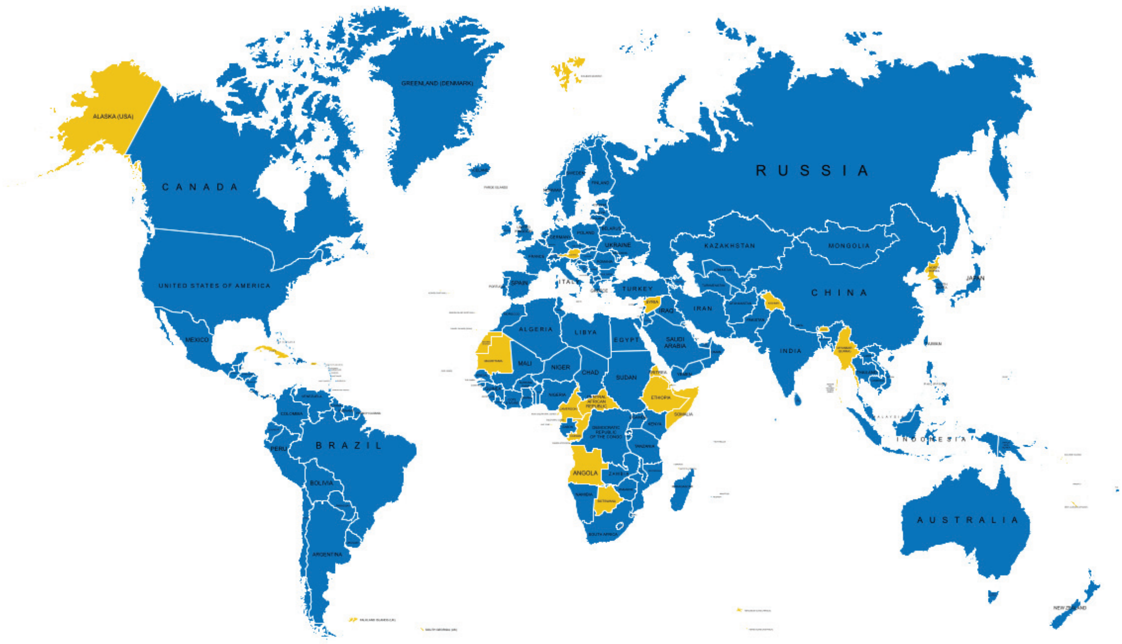


Figure 13 – The Coverage (in blue color) of the Teorema Platform

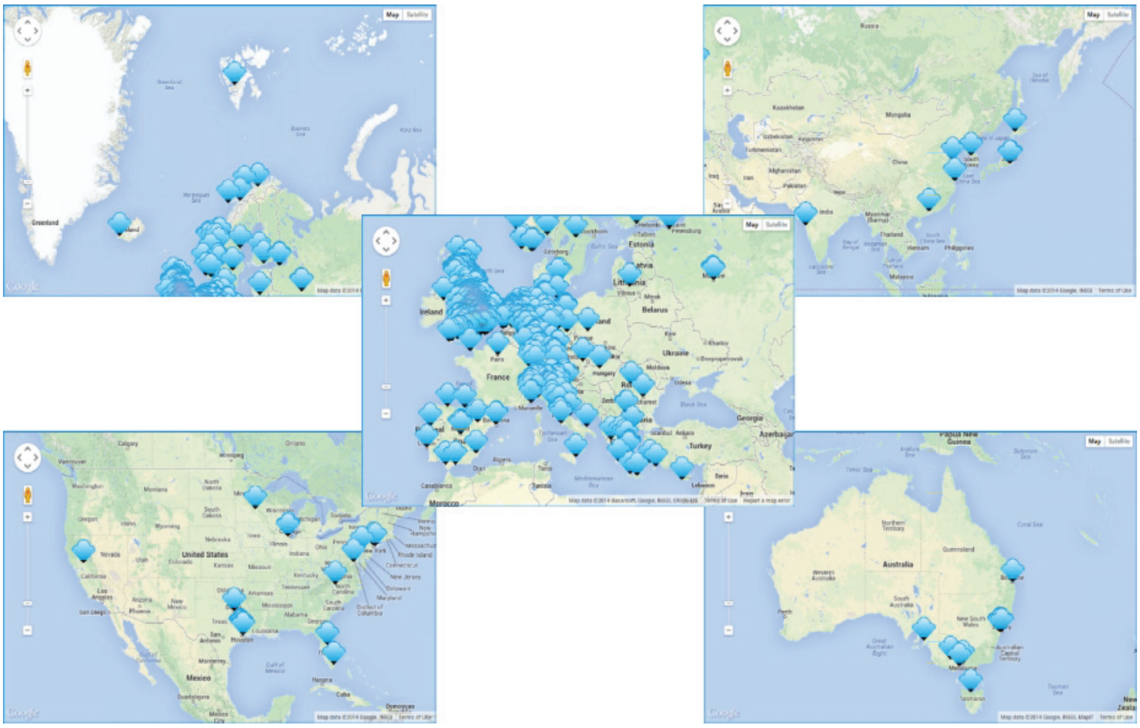


Figure 14 – The Installed Base Clusters of the Teorema Platform

Overall, the adoption of Teorema led to very significant savings. In the last quarter, around 20% of the assistance interventions for Teorema-controlled machines have been performed remotely (i.e., phone-fix). This has brought a reduction in maintenance costs of almost 25%, and Carpigiani estimate that, in the long run, reductions could stabilize around 40%. From the Carpigiani's customers' perspective, the prompt improvement in maintenance intervention enabled by Teorema translated into much smaller machine downtimes and, therefore, significantly decreased loss of revenues due to faults.

In other words, Teorema have already proven the importance of keeping the machines under real-time control and promptly identifying faults.

In addition, in offering a full range of innovative services, Teorema helps Carpigiani Customers to improve their workflow, avoid downtime, ensure the machine works at its very best and then to operate more efficiently.

Teorema platform provides Carpigiani customers with four new service packages, a complete cleaning program plan, and a reporting service for the real-time access to the production data of their machines (e.g., total ice cream dispensed, daily ice cream dispensed by flavor, alarms generated, hours of work performed by the motors, pasteurization cycles completed, etc.) which represent a disruptive innovation in after-sales services within the ice cream machines market.

Chapter 4

Related Works

4.1. Introduction

The main works that relate to Carpigiani e-Maintenance solution, applied or proposed in different industries, are reviewed in the following.

4.2. e-Maintenance Platforms

An e-Maintenance platform or infrastructure consists of software, hardware, and new technologies allowing to offer a certain service (Muller, Crespo-Màrquez, and Iung, 2008). A first classification of the e-Maintenance platforms can be done as follows:

a. HW/SW platforms (i.e., ICAS, WelCOM);

b. Platforms implementing e-Maintenance software capacities (i.e., PROTEUS, DPSA);

c. Software suite to support full e-Maintenance software capacities (i.e., CASIP);

d. Platforms for research and education (i.e., TELMA);

e. Other on e-Maintenance (i.e., DYNAMITE, PROMISE, SMMART, WSDF).

Among these platforms, the HW/SW and full platforms support mainly an evolution of the condition-based maintenance to which is added access to remote experts using Web interface for decision-making aid. Only a few platforms (i.e., TELMA, DYNAWeb) try to be fully consistent with the global e-Maintenance philosophy (e.g., concept of new services) (Levrat, Iung, and Crespo-Màrquez, 2008).
Each platform is described briefly below.

4.2.1. Integrated Condition Assessment System (ICAS)

The Integrated Condition Assessment System (ICAS) is one of the first HW/SW e-Maintenance platform developed. In fact, since the early 1990s, the United States Navy has been augmenting ship control by developing ICAS.

The system is designed to provide a computerized engineering tool to implement Condition Based Maintenance (CBM) via data acquisition, historical trending, expert analysis to improve equipment availability and reliability. ICAS utilizes an expert system shell running on networked workstations residing in each major machinery compartment.

The software ICAS is a Commercial Off the Shelf (COTS) product developed by IDAX Inc. for which the U.S. Navy holds Government Purpose License Rights (GPLR). It is a configurable, shell type architecture to allow for varied implementation of machinery monitoring and Condition Based Maintenance.

ICAS was installed on over 100 U.S. Navy vessels and land based sites. A typical U.S. Navy installation consists of four or five workstations, one in each major machinery compartment, connected by an active local area network (LAN). Each workstation is equipped with a Configuration Data Set (CDS), which contains the engineering information (DiUlio et al., 2003).

Besides, ICAS can be integrated to the Machinery Prognostics and Diagnostics System (MPROS) developed by a consortium led by Honeywell (PredictDLI, Knowledge Partners of Minnesota, the Georgia Institute of Technology, York International, and WM Engineering) for diagnostic and prognostic of the ship machinery. The system consists of MEMS and conventional sensors on the machinery, local intelligent signal processing devices, and a centrally located subsystem which is designed so that it can run under shipboard monitoring systems such as ICAS. The MPROS collects data continuously and the various algorithms of diagnostics and prognostics work together to produce the analysis (Hadden et al., 2003).

Comparing with Teorema, it is observed that ICAS does not enable the management of large scale and low-cost solutions due to the complex architecture and the need of LAN and workstations.

4.2.2. Distributed Prognosis System Architecture (DPSA)

The National Aeronautics and Space Administration (NASA) has proposed a distributed architecture for Prognostics and Health Management (PHM) called Distributed Prognosis System Architecture (DPSA).

The anomaly, diagnostic, and prognostic technologies implemented at the lower levels (Line Replaceable Units, i.e., LRUs), and subsystem levels are used to detect and predict off-nominal conditions or damage accumulating at an accelerated rate. This information is then analyzed through the hierarchy of reasoners to make informed decisions on the health of the systems and how they affect total capability and remaining useful life.

NASA, the U.S. Department of Defense (DoD), and industry have been applying this technology for use in their next-generation vehicle health management solutions (Roemer et al., 2006; Roemer et al., 2011).

4.2.3. Web-Services-based e-Diagnostics Framework (WSDF)

To fulfill the requirements of e-diagnostics systems and remedy the shortcomings of existing diagnostic systems, a Web-Services-based e-Diagnostics Framework (WSDF) had been developed by the Department of Electrical Engineering of the Chung Cheng Institute of Technology (Taiwan). WSDF is based on modern Web-services technology (XML, SOAP, etc.) and it can achieve the automation of diagnostic processes and information integration for semiconductor equipment (Hung, Cheng, and Yeh, 2003).

WSDF solves both security and communication problems in the e-diagnostics system. WSDF is intended to support the e-diagnostics functions defined by the international nonprofit consortium SEMATECH (http://www.sematech.org) that performs research to advance chip manufacturing. The e-Diagnostics capabilities are defined by four levels, i.e., Access and Remote Collaboration, Collection and Control, Analysis, Prediction.

It is believed that WSDF can be applied the SEMATECH model and to construct e-diagnostics systems for semiconductor manufacturing industry.

Compared to Teorema platform, WSDF seems to be a consistent but highly vertical solution-set and solely oriented to achieve diagnostic for semiconductor equipment.

4.2.4. PROTEUS Platform

This platform is based on the results from the project PROTEUS sponsored by the French Ministry of Economy, Finance and Industry and the Federal Ministry of Education and Research of Germany under the label of European Commission Initiative ITEA.

The basic integration technology as defined for the PROTEUS platform allows for the exploitation of Web-services to access different systems, among them there are those from the Manufacturing Execution System (MES) level or condition monitoring equipment. Each adapter used to interface an individual tool provides an object-related view of the relevant data of this tool.

The original idea of the PROTEUS project dedicated to industrial maintenance lies in the integration of all the necessary tools whose functions range from the detection of alarms to the management of spare parts, with the purpose of optimizing costs and improving productivity. This optimization can be seen as the extension of the automatic control principles throughout the entire enterprise, in particular the "closed loop" concept applied to the production process.

Indeed, maintenance is a very important activity for all industrial enterprises which needs the integration of several sub-systems associated with the different previous functions involved in service operations. All these software sub-systems are currently based on different models; they are normally complementary, but sometimes redundant, incoherent and always heterogeneous.

The objective of PROTEUS is then the integration of these various sub-systems thanks to a unique and coherent description of the equipment, a generic architecture and coherent models of heterogeneous components (Bangemann et al., 2006).

It is then possible to access all the documentation in a unified way, independently of its origin, equipment manufacturer, integrator, or end-user. Any user also has the capability to remotely access relevant data from the plant, to request a diagnosis from a remote intelligent system, and then to prepare any intervention with optimality.

Such a platform may be adapted to any production tool, in any domain (energy produced in a nuclear power plant or by windmill, oil and gas and heavy industry applications, discrete part manufacturing companies, transport systems, etc.). It allows any maintenance strategy, allows the outsourcing of maintenance to dedicated and specialized companies, and the optimization of the different tasks or operations for the benefits of the different actors.

PROTEUS is mainly focused on the infrastructure for integration rather then on the development of dedicated tools. Considering the market situation diverse solutions are available for all the components arranged around the central operational core generating a great range of application, e.g., an instantiation of the generic PROTEUS platform to ship maintenance should be able to use ICAS tool for both data acquisition and data analysis.

4.2.5. PROMISE Project

The PROduct lifecycle Management and Information tracking using Smart Embedded systems (PROMISE) project focused on gathering, processing and delivering relevant product information during the complete lifecycle of the product, addressing existing gaps in the information flow and creating better understanding of total life cycle events and the total costs of products over the lifecycle.

PROMISE started in 2004 as a European Union funded project comprising a consortium of 22 industrial and academic partners. The Project ended in May 2008 when it was officially acclaimed by the European Commission as an example of a successful project.

By feeding product critical information back to the earlier design and production phases and forwards to the appropriate intervention areas PROMISE gives key players effective instruments to create better services and add value to the composite product.

The breakthrough contribution of PROMISE is to allow information flow management to go beyond organizational barriers, securely. It effectively closes the product lifecycle information loops using the ICT, and seamlessly transforms that information into knowledge.

New generation Information Tracking and Flow Management systems allow all actors involved in the product's lifecycle to track, manage and control lifecycle information at any phase of the product lifecycle at any time and from any location.

To address these issues, within the project, a toolbox of algorithms for multisensor performance assessment and prediction, named Watchdog Agent, has been developed at the Center for Intelligent Maintenance Systems (IMS) which is a multi-campus NSF Industry/University Cooperative Research Center between the University of Cincinnati and the University of Michigan. This tool can be utilized to realize predictive condition-based maintenance as well as to identify the components that possess significant remaining useful life that could be efficiently and cost-effectively disassembled and reused. A wide variety uses for the Watchdog Agent have been devised to address many applications of a different nature, with different levels of complexity and criticality. Additionally the Watchdog Agent is integrated with the IMS Device-to-Business (D2B) platform that provides a link between the shop floor and e-business (Lee et al., 2006; Djurdjanovic, Lee, and Ni, 2003).

4.2.6. Dynamic Decisions in Maintenance (DYNAMITE)

In parallel of PROMISE, the new European project Dynamic Decisions in Maintenance (DYNAMITE) aimed at creating an infrastructure for mobile monitoring technology and delivering new devices which will make major advances in capability for decision systems incorporating sensors and algorithms. The objective was to deliver a prototype system with a clear exploitation route to take the technology into the European market.

The information and communication infrastructure is referred here as DYNAWeb e-Maintenance services and it is based on OSA-CBM standard.

In addition, a novel Maintenance Decision Support System (MDSS) was developed that offers different strategies for cost-effectiveness that can be applied integrated or separately (Holmberg et al., 2010).

The DYNAMITE Project was an European Community funded research project and was completed in February 2009 and was coordinated by VTT Technical Research Centre of Finland.

4.2.7. System for Mobile Maintenance Accessible in Real Time (SMMART)

Launched in November 2005 and programmed on a three years period, System for Mobile Maintenance Accessible in Real Time (SMMART) is a European research and development project run by a consortium of 24 partner companies from 10 European countries. Turbomeca (http://www.turbomeca.com) is the project coordinator.

Designed to the transport industries (aeronautical, road, and maritime), SMMART has developed and demonstrated a new integrated information system concept, for fleet maintenance. SMMART is used to remotely gather information concerning the fleet in service in real time, process the data in a central database, and share this data with all the support actors worldwide.

The innovative approach of the SMMART consortium was based on Fraunhofer s-net wireless protocol stack, which requires only a tiny amount of energy to initialize and operate a wireless multihop sensor network. The combination of this new technology with smart tags and MEMS is a solution able of operating and communicating wirelessly in the harsh environment of a vehicle's propulsion unit to monitor usage and maintenance data throughout the life-cycle of critical parts and provide secure end-to-end visibility of the logistic supply chain.

Business process change support will enable European companies to offer new more flexible and customized support and logistics' services for transport industry (Zephir, and Minel, 2007).

4.2.8. Computer Aided Safety and Industrial Productivity (CASIP)

The Computer Aided Safety and Industrial Productivity (CASIP) is a distributed software solution implementing remote diagnosis and full e-Maintenance capacities.

CASIP integrates both the modules of design allowing to analyse deterioration, their causes, effects, and symptoms (using tools such as FMECA, HAZOP, Fault-Tree Analysis) and the modules of Proactive Maintenance (i.e., Monitoring, Diagnosis, Prognosis).

CASIP complete the Enterprise Resources Planning (ERP) to manage the risk of the plants and the Manufacturing Execution System (MES) to react in real-time regarding to malfunctioning. At the ERP level, CASIP is open to Computer Aided Manufacturing (CAM), Computerized Maintenance Management System (CMMS) and other existing systems. At the MES level, CASIP is open to Supervisory Control and Data Acquisition (SCADA), Programmable Logic Controller (PLC), Distributed Input/Output, Data Acquisition System and other control systems. CASIP-SAM is the module of CASIP which supports concretely remote monitoring, remote diagnosis and remote maintenance for multi-site companies.

CASIP-SAM integrates a real-time database allowing acquiring continuously the data on the process, processes to react at a first level and a permanent database ensuring the historic and the exploitation of knowledge expertise. This application can detect malfunctions, make diagnoses based on recorded cases of risky situations, and predict future evolutions in the system. To do this, CASIP-SAM is structured on three analysis levels:

a. reactive level, connected to the communication networks, for real time data acquisition;

b. proactive level, for data processing and storage (e.g., ORACLE, SQLServer, etc.);

c. a remote level for distant access over the Internet to/from anywhere.

The integration of the TCP/IP protocol allows CASIP to communicate with the Enterprise Management level (ERP) and the Operational level (MES), and this confers a large range of possibilities to interfacing as well as distributed Input/Output. An independent communication also ensures robustness and autonomy in particular in the event of failure of the Supervisory Control.

Moreover, CASIP integrates XML in order to get an ergonomic Man-Machine Interface with ample software portability. An XML (eXtended Markup Language) interactive software is generated to get a wide portability of the Diagnostics Expert System which can be used on an Internet Browser (Levrat, and Iung, 2007).

CASIP is commercialized by Predict (http://www.predict.fr) under the name of Knowledge and Advanced Services for e-Maintenance (KASEM).

4.2.9. TELMA Platform

The TELMA platform is located at the Henri Poincaré University of Nancy (France) and was developed for supporting research and education. Thus, the platform is designed for a local use in the frame of conventional training activities, a remote use via Internet for operation on industrial e-services, and for accessing to performance data.

From a research point of view, the platform is currently used to demonstrate the feasibility and the potential benefits of approaches in relation to e-Maintenance (i.e., within DYNAMITE project). The maintenance part of the platform is built on the CASIP (KASEM) product supporting a local real-time maintenance system, a centralized maintenance system (with Oracle) and some remote stations . To this maintenance system is integrated through SQL-Server and Oracle, a lot of hardware and software components for running the maintenance actions well such as OPC server, a CMMS called EmpaciX, a Technical Data Base System called Advitium, an ERP called AdoniX, etc. The engineering and deployment of CBM and proactive maintenance strategies is consistent with OSA/CBM proposal functionalities (Levrat, and Iung, 2007).

This platform is currently fully operational. Students and teachers use it for maintenance purpose either on site or on a remote way through the VPN.

4.2.10. WelCOM Project

The Wireless sensor networks for Engineering asset Life Cycle Optimal Management (WelCOM) project is a concerted effort that aims to facilitate the implementation of a CBM strategy within an e-Maintenance framework, employing a flexible toolset of software and hardware solutions based on Wireless Sensor Networks and Mobile Devices.

This research is conducted by a Greek public-private partnership between the ATHENA Research & Innovation Centre and the NHRF public research institutions and Kleemann Hellas Lifts, Prisma Electronics, GDT, and Atlantis Engineering as industrial partners.

The WelCOM platform utilizes a smart sensor infrastructure for machinery condition monitoring, while interfacing with a CMMS to deliver a context aware asset management tool. The platform offers multi-layered intelligence for the optimal operation and maintenance of the equipment. At the device level, a new optical sensor is designed in both wireless and wired form so that it can be used for diverse monitoring applications.

The optical sensors are integrated in networked embedded systems forming a wireless sensor network that can sustain operation in industrial settings. The wireless sensor network nodes implement a distributed sensor-embedded intelligence based on condition state models and a novelty detection engine. The overall system integrates real-time condition monitoring and novelty detection with accurate fault diagnosis; support of prognosis and prediction of failures; and effective notification for immediate attention and maintenance.

The maintenance data can be provided by heterogeneous sources; hence data integration is a major obstacle for wide adoption. To this end, maintenance information modeling will be utilized and based on open specifications and on the MIMOSA OSA-CBM standard. Also, data interoperability and exchange formats and interfaces have been defined so as to allow seamless information flow between the wireless sensor network layer and the knowledge management layer.

The WelCOM configuration and management portal will offer all authorized personnel access to real-time, on-the-spot sensor feeds of modelled machine state data, while traditional CBM approaches allow only delayed, off-line processed information. The portal will couple services and sensor software components as to enable checking from the administration level down to managing and calibrating any single sensor device of the monitoring network.

The knowledge produced within the platform is not only available in the control room. A significant component is the portable intelligent maintenance advisor. This module is available as a software for portable devices that scans through the knowledge portal and retrieves proper information to aid the practice of a technician in the field.

The architecture is being validated in piloting projects on electric elevator currently at Kleemann's Lift Research Tower.

The WelCOM intelligent advisor should be able to monitor, map and analyze a set of condition parameter related to lift operational parts and components (Pistofidis et al., 2012).

Finally, a training module will be integrated into the WelCOM platform aiming to achieve a defined level of competencies for maintenance technicians by harnessing the derived knowledge. The e-training is a practical and cost-effective alternative to expensive standard training (Papathanasiou et al., 2012).

It shall be noted that it is difficult to compare Teorema platform with WelCOM and most of these e-Maintenance platform projects because they were all deployed and evaluated in a lab environment and there is a significant lack of large-scale real-world applications and informations.

4.3. e-Maintenance Applications

The domain of industrial systems is increasingly changing by adopting emerging Internet-based concepts and technologies.

e-Maintenance applications supporting full e-Maintenance capacities are described below.

4.3.1. Canon eMaintenance

eMaintenance is a new service concept from Canon (http://www.canon.com) to manage and control all multifunctional devices. eMaintenance is based on Canon's *Remote Diagnostics System* (RDS) which gives the information about connected machine via Internet and, allows the central administrator to monitor the status, workload and usage pattern for each individual device.

The eMaintenance system consists of a secured Internet server (*Universal GateWay*, or UGW) and monitoring equipments. These monitoring equipments collect the device information and send it to UGW.

Three types of monitoring equipments are available: *embedded RDS (eRDS)* (system capturing information from devices), *RDS Agent* (hardware designed to monitor 1 to 30 devices), *RDS Server* (server designed to monitor up to 3000 devices, subjected to network environment).

The system will send an alert if a critical fault occurs, or when a device is not performing up to standards (e.g., if paper-jams exceeds a given value, the service provider will be alerted by email). Statistics will be automatically collated and sent to the administrator for the provision of timely support. eMaintenance includes consumables management (e.g., email notification when toner is running low) (Chowdhury, and Akram, 2011).

eMaintenance implements high security standards by design. All communication between the system and Canon devices are encrypted using HTTPS protocol on TCP port 443.

The system automates firmware download making it possible to update entire device fleet in a shorter time frame. Updates can be scheduled to take place during off-peak hours to further reduce interruptions.

The eMaintenance system is designed to minimize network load and data traffic. The data transmitted for each event are approximately 4 Kbytes. The monitoring equipment collects approximately 3MB a day per device, depending on device type, setting, and frequency.

The commercial proposal consists of 3 different packages: *eMeter-reading* (system providing automatic service billings), *eToner* (system managing toner-on-demand), and *eProactive* (intelligent system performing remote diagnostics).

An analysis performed on patents filed by Canon in the period 1990-2013 (year of priority), on e-Maintenance solutions, found 593 patent families equivalent to 1979 patent applications.

Canon eMaintenance is very similar to Teorema concept but limited to a specific type of equipment (i.e., multifunctional devices) and to a defined geographical area (for now).

4.3.2. Alstom TrainTracer

TrainTracer is an on-board train data system that continuously collects data through sensors and relays it to a ground-based server by remote transmission. This information available in real time or upon request allows to analyze the situation on board, and anticipate problems or troubleshoot failures. TrainTracer is widely used by Alstom Transport (http://www.alstom.com) in its maintenance service but also provides feedback to significantly raises effective fleet availability.

Alstom has implemented TrainTracer on the UK West Coast Main Line fleet of 53 Pendolino, as a fully integrated service in operator Virgin Trains and maintainer Alstom's everyday operations. As a result of robust marketing and efficient TrainTracer operation, the number of Virgin Trains customers using the West Coast Main Line rose from 13.5 million to 22 million in four years. In fact, combined with GPS, TrainTracer module provides the operator with precise information about where their trains are located, what speed they are moving at, whether they are respecting their timetable and what their energy consumption is at a given point of the route or for a given driver. The trains are also able to detect vertical or lateral infrastructure movements. The statistics showed that most of the delays were due to infrastructure problems. For this reason Virgin Trains has begun using TrainTracer to monitor and supervision the condition of the track.

Alstom were innovative with their business process too. They created an organization called Train Life Services, and offered the Pendolino trains to Virgin on availability contracts that may extend beyond 20 years. If Alstom only had focused on manufacturing and selling trains, they would not have attracted Virgin's business.

It shall be noted that similar incidents have occurred in other industries, e.g., Rolls-Royce were motivated to develop their power-by-the-hour service by American Airlines (Baines, and Lightfoot, 2013).

4.3.3. Falcon eMaintenance

Leading aerospace manufacturer Dassault Falcon (http://www.dassaultfalcon.com) completed successful testing with Falcon operators based in Europe and the United States on a new maintenance program called Falcon eMaintenance. The program allows the Dassault Falcon Customer Service to remotely access an airplane Central Maintenance Computer, or any other aircraft maintenance application, to troubleshoot and diagnose issues. This connection is made through a PC provided by Dassault with WiFi and satellite capabilities.

The maintenance system makes it possible for the aircraft technicians, service center and the Falcon Technical Center to work in real time on one common communication platform that is capable of both audio and video. From there, they can share technical documents, transfer files and organize multimedia conference meetings. If needed, it will be possible for an Authorized Service Center or the Falcon Technical Center to remotely access the aircraft's central maintenance computer (Chowdhury, and Akram, 2011).

Dassault Falcon has then shift from standard maintenance operations to FalconCare program based on a pay-as-you-fly philosophy.

4.3.4. Joy Global Smart Services

Joy Global (http://www.joyglobal.com) produces mining equipment and parts, and provides services. Increasingly, this equipment has been smart and networked allowing Joy Global to automated mines, improving safety and productivity. Smart, connected products are also the core of Joy Global's Smart Services offering, which the company considers one of its primary competitive differentiators.

Joy Global monitors operating conditions, safety parameters, and predictive service indicators for entire fleets of equipment far underground, and analyzes product data to predict potential failures in real time on components.

By fixing equipment before it breaks and prioritizing which maintenance activities to conduct based on performance and expected failure, Smart Services reduces costs and minimizes downtime for customers. Joy Global also uses data drawn from machines operating across its global network to compare mine processes against benchmarks. This prognostic engine analyzes data to reveal process deficiencies and recommend corrective action through operator training, system operating parameter adjustments, and machine changes.

Mining equipment can now operate autonomously and this significantly increases safety and improves mine productivity.

As Joy Global's product capabilities have evolved, the company has gradually changed its business model from selling products to selling products and services. The basic service offering consists of spare parts distribution, courses that train mine personnel on optimizing equipment operations and maintenance, on-site support by field service engineers, and the capability to repair and rebuild machines back to OEM specifications. Customers that want Smart Services must upgrade their equipment to smart, connected machines and agree to share machine data with Joy Global.

The Smart Services offering has been a first step towards new business model offerings, such as performance-based service contracts that include guaranteed uptime where Joy Global shares risks and rewards with customers through payments based on the performance of the equipment and the output of the mine.

It shall be noted that companies whose products (and associated technological capabilities) are central to overall product system operation and performance, such as Joy Global's mining machines, will be in the best position to enter related products and integrate the system. Manufacturers that produce less system-critical machines, such as the trucks that move the material extracted from underground, will have less capability and credibility to take on a broader system provider role (Porter, and Heppelmann, 2014).

4.4. Vendor Solutions

To date there are few commercial platforms that can handle full e-Maintenance software capacities. The main solutions are briefly described below and compared between them in order to have an overview.

4.4.1. Parametric Technology Corporation

Parametric Technology Corporation (http://www.ptc.com), founded in 1985, is a leading software company specializing in 2D/3D Computer-aided design (CAD) software and product lifecycle management (PLM). Parametric Technology Corporation (PTC) employs approximately 6,000 professionals serving more than 28,000 businesses in manufacturing industries worldwide. Recently, PTC quickly scaled to a dominant position of IIoT leadership through its acquisitions of ThingWorx and Axeda.

ThingWorx (http://www.thingworx.com) is the first platform designed to rapidly build and run connected applications in the IoT. Essentially, it provides the conversion layer enabling developers to translate the data collected from sensors and devices into information for various business applications. By reducing complexities in application development, the ThingWorx platform significantly simplify new developments. For example, the Mashup Builder provides a drag and drop interface, allowing developers to build HTML-based applications without writing complex codes.

Axeda (http://www.axeda.com) provides software and cloud-based services for managing connected products and machines and implementing innovative M2M and IIoT applications. Axeda requires the use of a proprietary messaging protocol (Axeda Wireless Protocol, i.e., AWP) and provide SOAP and RESTful web services to facilitate the development of applications. The platform features remote device diagnostics and repair, as well as dashboards, reports, and proactive fault detection.

On January 29, 2015, PTC announced ThingWorx-Axeda Integrator, which unifies the ThingWorx development platform with the Axeda connectivity, device cloud and applications, providing a complete IoT solution stack for the industry.

The ThingWorx-Axeda Integrator allows Axeda machine data to be immediately available in the ThingWorx platform, and Axeda APIs to be accessed from ThingWorx Composer rapid application development tool. The Integrator also provides synchronized model and data between the two systems. Data items, alarms, and mobile locations flow into the ThingWorx platform via Axeda's Machine Streams data feed service. Machine Streams can send date natively directly into ThingWorx.

In parallel, leading asset health software solution company Mtell (http://www.mtell.com) has announced it has joined PTC, as a ThingWorx Partner. The combined solution leverages the Mtell Machine Learning Technology in conjunction with the ThingWorx Rapid Application Development platform to create Advanced Predictive Analytics for IoT applications. The Mtell Industrial Internet uses machine learning to gather and analyze Big Data from networked sensor signals to share learned behavior between industrial machines and adjust operations and maintenance activities.

4.4.2. Eurotech Everyware

Eurotech (http://www.eurotech.com) Everyware platform simplifies device and data management by providing a solution that integrates distributed devices with business enterprise applications.

Eurotech develops technical building blocks required to assemble distributed systems of devices and sensors which are effectively connected to IT infrastructures. The Everyware solution is a combination of hardware, firmware, operating systems, programming frameworks that allows the implementation of M2M and IIoT projects on a read-to-use infrastructure. Everyware is built on the open standard TCP/IP-based protocol MQTT.

Everyware Cloud (EC) is the proprietary cloud infrastructure that, in conjunction with the Everyware Software Framework (ESF) programming environment, enables Eurotech devices to send and receive data to the end application.

Everyware Device Cloud (EDC) is the combination of all required building blocks for designing, implementing, deploying and running device to business application infrastructures. EDC integrates hardware platforms, operating systems, ESF, EDC clients and the EC into one single system.

While Everyware Cloud is Eurotech's public cloud offering, customers can use the Everyware Server (ES) appliance for on-premises deployment. The ES appliance can also be used as a local aggregator of distributed devices and sensors.

The most characteristic aspect of Eurotech’s Everyware Cloud platform is the integration of the different elements required to build a complete end-to-end connected device solution. Eurotech considers that the intersection of these technologies can creates a differentiated business model.

4.4.3. ARM mbed and SaleForce

On October 2014, in San Francisco at Dreamforce 2014, was explain how to use ARM mbed and the Force.com REST API to build Salesforce IoT applications.

Salesforce Inc. (http://www.salesforce.com) is a global cloud computing company headquartered in San Francisco known for its Customer Relationship Management (CRM) product. Forbes magazine has ranked it the most innovative company in America every year since 2011.

ARM mbed (http://mbed.org) is a platform for internet-connected devices based on 32-bit ARM Cortex-M microcontrollers. The project is collaboratively developed by ARM and its technology partners (SalesForce, IBM, etc.).

The commercial proposal consists of 3 different software packages.

Firstly, the mbed OS event-driven operating system for ARM Cortex-M based devices that is designed specifically for energy constrained environments. The OS includes the connectivity, security and device management functionalities required in IIoT device. It is a full-stack OS, consolidating the complex but fundamental building blocks that are no longer an option in production devices. The mbed OS provides support for key standards such as Bluetooth Smart, Thread, Wi-Fi, and 802.15.4/6LoWPAN along with TLS/DTLS, CoAP, HTTP, MQTT and Lightweight M2M. The mbed OS takes up a scant 256KB of memory or less so it can be installed on tiny devices such as sensors. It includes C++ programming interfaces, an event framework, a communication manager and support for cryptography, among other things.

Secondly, the mbed Device Server licensable software that provides the required server-side technologies to connect and manage devices in a secure way. It provides a bridge between the protocols designed for use on devices and the APIs that are used by web developers. This simplifies the integration of devices into cloud frameworks that deploy Big Data analytics on the aggregated information. Built around open standards, the device server scales to handle the connections and management of millions of devices.

Finally, the mbed Tools, i.e., command-line build, component management and test tools that provide the platform toolkit. They also include a Web IDE, connectivity APIs and cloud build services.

4.4.4. Xively

Xively (http://xively.com), formerly Cosm and Pachube, was created in 2007 as a data brokerage platform. The name Pachube, pronounced Patch Bay, i.e., switching board as used in telephone operation centers, intended to state the scope of the project.

Like the telephone operator that connects people, Pachube allows mashing up data streams to build new applications.

Pachube became known during the nuclear accident in Fukushima (Japan) on 2011 when many Japanese used low-cost Geiger counters to monitor the radioactive fallout across the country. The data was streamed from Pachube to a live map using Google Maps API.

The business model is mainly around service and consulting, but also based on the frequency of sending and receiving data to and from the platform. Ownership of the information remains to the user, but the data is stored on a Xively server.

Xively provides a multitude of tools to help individuals and companies to build and manage connected products and applications based on connected things. There is an extensive list of API wrappers for different programming languages and platforms. This allows to build embedded software (C, ARM mbed, Arduino, etc.), cloud applications (Java, Python, Ruby), web apps (JavaScript, PHP), as well as smartphone apps (Android, Objective-C) that easily connect to the Xively service.

The Xively web service itself allows to provision, activate and manage devices. In essence this means to give each device a unique identity and specific rights to create and receive data on the platform. Besides, Xively has integrated triggers which can call a web service if, e.g., a threshold in a data stream was exceeded. Most of the development tools around Xively are open source (BSD 3-clause license) and hosted on GitHub.

4.4.5. Comparative Analysis of Vendor Solutions

The first characteristic to be analyzed is the integration of sensor technologies, i.e., the types of devices that are supported by the platform.

Platforms that require a proprietary gateway to connect devices are limiting their adoption by users and the reactivity to implement new protocols and support an increasing number of sensors (e.g., some solutions only enable RFID technology).

The surveyed solutions have addressed this issue differently. For example, the Axeda platform proposed a connection of sensors via a proprietary gateway (realTime.io which also requires the use of a proprietary transport protocol, ioDP), while the ThingWorx platform uses web sockets, MQTT or other communication protocols to interconnect devices to the platform. Eurotech Everyware always needs a gateway. Xively full supports heterogeneous sensing and actuating devices.

It is essential to establish standardized protocols for all devices, as it is currently done for highly constrained devices by the IETF (Ishaq et al., 2013). At present, constrained protocols are supported by ARM mbed (MQTT, CoAP). The others assume the use of relatively powerful devices capable of supporting traditional web protocols.

A second characteristic to be analyzed is the type of architecture. All platforms considered are provisioned from a cloud (i.e., cloud-based) in a form of a Platform-as-a-Service (PaaS). The PaaS refers to the platforms that provide cloud computing services for devices and data. The services includes, but are not restricted to storage facilities, devices management, devices connectivity, backup mechanisms or online support. It shall be noted that Axeda, ThingWorx and ARM mbed are machine-to-machine (M2M) oriented while Xively is more dedicated to a deployment of sensors and actuators (e.g., Smart Manufacturing).

Regarding the support for the development, all the platform are using nonuniform REST APIs and data models which makes difficult the mashing up of data. Xively APIs can be accessed via transports such as HTTP, MQTT, and Sockets.

Xively provides open source libraries (but not the platform); all the others platforms offer proprietary libraries with only basic functionalities (e.g., connection to the platform with access keys). Axeda platform provides a full Software Development Kit (SDK) written in Groovy for application developers.

Finally, only Axeda enables a marketplace, i.e., a catalog wherefrom applications or services could be purchased, such as Apple App Store or Google Play.

4.5. Summary and Concluding Remarks

As presented in the previous chapters, e-Maintenance is an interdisciplinary subject with multiple links. Therefore, the related works of potential interest are a huge amount. The works presented were searched and selected based on their relevancy to be fully consistent with the global e-maintenance philosophy.

Most of the e-Maintenance Platforms analyzed were deployed and evaluated in a lab environment and there is a shortage of large-scale real-world experiences. Proprietary e-Maintenance Applications and Vendor Solutions supports very well the after-sales activities of OEMs but none of them consider the point of view of a service company. Indeed, these solutions does not take into full account the intrinsic variability of the real environment where a traditional service industry can operate. Several service platforms are based on the concept of remote control of the system (or machine) to be diagnosed, working in homogeneous environment and needing for additional resources. These systems are not compatible with one another and their complexity and lack of well-known tools prevent a wider dissemination outside of a relatively restricted community of expert users. It shall be noted, in fact, that a freelance service company performs interventions (e.g., installation, diagnosis, maintenance) on a variety of systems. For example, in a restaurant may be present a plurality of equipment from different manufacturers. There can be multiple suppliers for the same equipment and there may be the need to change some of them. In other words, it works in an actual and dynamic environment where it would be very difficult to implement the proposed architectures in a proper way.

All this brings us back to the issue of standards and communication protocols.

Chapter 5
Conclusions

Internet of Things are not limited to the Internet and data centers. Indeed, in the increasingly connected future, many commercial and consumer devices will be part of distributed systems. Traditionally, automated monitoring, notification, and adaptation have not been extended to these systems if they were not part of critical infrastructure. This is likely to change as we rapidly approach the Internet of Things.

The recent ICT advances support and foster the realization of large-scale e-Maintenance solutions for the remote monitoring and assistance of household and similar appliances.

The decrease in costs will soon enable the realization of e-Maintenance platforms, which are now affordable both for (at least) for higher-end appliances, such as food-processing machines, and also on lower-end appliances, such as households.

This thesis demonstrates how it is possible to realize the e-Maintenance of Smart Connected Machines in the Industrial Internet of Things. The experience with Teorema demonstrates that large-scale e-Maintenance platforms are already mature for the adoption in mass-produced equipment. We expect that, in the future, e-Maintenance solutions, such as Teorema, will be investigated and put in place for other household and similar appliance markets.

A first important lesson brought out by the present research is the potential of Open Source technologies, already mature for the industrial Internet use and very well supported. In fact, Open Source technologies reduce development costs, protect the investments by not restricting the software platform to a single proprietary vendor, and present several advantages such as rapid application development, thanks to many components already available (and to a collaborative Internet community).

It shall be noted that applications in manufacturing usually were based on proprietary solutions (often built from scratch, with no reuse allowed, no platform and technologies to build upon, i.e., very expensive).

Another important lesson learned is the growing importance of the role of ICT and Internet ecosystem (M2M, Mobile, COTS SBCs, COTS Open Sources Software, Cloud Computing, Big Data, Security Solutions) in the digital manufacturing.

This new wave of IT-based innovation is having a disruptive impact on competition and strategy. In the 1960s and 1970s, IT brought automation to a number of discrete business processes. Then in the 1990s, the connectivity and Internet enabled companies to integrate processes, as well as develop online services. The first two IT waves helped companies become more productive by reengineering their operations, but their products were unaffected. The third wave is changing everything. Pervasive networks of mobile things embedded with innovative sensors are now revolutionizing manufacturing and products and trasforming competition.

This work has greatly influenced the R&D strategy of Carpigiani Group and it will continue to be developed in the future.

Research directions for future works will be standards and protocols study for the IIoT. The proprietary protocol choice was made because no mature M2M procotol were available when Teorema was designed. As Carpigiani's ice cream making machines will increasingly operate in a local interconnected environment, e.g., "smart kitchens", "M2M anywhere kitchen", "i-restaurant" or "on-line kitchens" structures, it is necessary that future versions of Teorema will switch to standard protocols as soon as they will be available.

New research will address the application of Big Data. In the Internet world, the term Big Data indicates tools, environments, algorithm, for the analysis and processing of huge amount of data. This could be useful for the (even real time) processing of machine data flows, e.g., for comparative analysis, and automated anomaly/fault detection.

Finally, I want to highlight the importance and value of this experience of "Industrial" Ph.D. for the author and for the Carpigiani Group.

Co-operation between higher education and business is now widely recognised to have benefits for both sides. It stimulates knowledge transfer in both directions and leads to long-term partnerships, accelerating product development and unlocking new opportunities for innovation. Also, it makes funding work more effectively in Education and Research.

The Teorema Project is an example of successful co-operation between the both sides. As a consequence of being involved with this research, Carpigiani Group and University of Ferrara will work together in the Smart Manufacturing 2020 Project (of the Cluster Tecnologici Nazionali call) in the WorkPackages “Smart Maintenance” and “Smart Product and Services”, that are based on novel e-Maintenance platforms.

In addition, Carpigiani Group and University of Ferrara will jointly submit a proposal for an EU Horizon 2020 funded innovation project based on the development of this work.

Chapter 6

References

Ahrens, J. P., Hendrickson, B., Long, G., Miller, S., Ross, R., and Williamson, D. (2011), "Data-Intensive Science in the US DOE: Case Studies and Future Challenges", *Computing in Science & Engineering*, Vol. 13, No. 6, pp. 14–24.

Alexander, F., Hoisie, A., and Szalay, A. (2011), "Big Data", *Computing in Science & Engineering*, Vol. 13, No. 6, pp. 10–12.

Ashton, K. (2009), "That 'Internet of Things' Thing", *RFID Journal*, 22 June 2009, pp. 97-114.

Bachle, M., and Kirchberg P. (2007), "Ruby on Rails", *IEEE Software*, Vol. 24, No. 6, pp. 105-108.

Baines, T. S., and Lightfoot, H. W. (2013), *Made to Serve: How manufacturers can compete through servitization and product service systems*, John Wiley & Sons.

Baines, T. S., Lightfoot, H. W., Benedettini, O., and Kay, J. M. (2009), "The servitization of manufacturing: A review of literature and reflection on future challenges", *Journal of Manufacturing Technology Management*, Vol. 20 No. 5, pp. 547-567.

Bandyopadhyay, S., Sengupta, M., Maiti, S., and Dutta, S. (2011), "A survey of middleware for internet of things", in *Recent Trends in Wireless and Mobile Networks*, Springer, Berlin Heidelberg, Germany, pp. 288–296.

Bangemann, T., Rebeuf, X., Reboul, D., Schulze, A., Szymanski, J., Thomesse, J.-P., Thron, M., and Zerhouni, N. (2006), "PROTEUS - Creating distributed maintenance systems through an integration platform", *Computers in Industry*, Vol. 57, No. 6, pp. 539–551.

Bryant, R. E. (2011), "Data-Intensive Scalable Computing for Scientific Applications", *Computing in Science & Engineering*, Vol. 13, No. 6, pp. 25–33.

Belqasmi, F., Glitho, R., and Fu, C. (2011), "RESTful Web Services for Service Provisioning in Next-Generation Networks: A Survey", *IEEE Communications Magazine*, Vol. 49, No. 12, pp. 66–73.

Bogers, M., Afuah, A., and Bastian, B. (2010), "Users as innovators: A review, critique, and future research directions", *Journal of Management*, Vol. 36, No. 4, pp. 857–875.

Brunello, G. (2003), “Microprocessor-Based Relays – an Enabler to SCADA Integration”, *Electricity Today*, Vol. 2003, No. 4, pp. 10–11.

Cai, N., Wang, J., and Yu, X. (2008), “SCADA System Security: Complexity, History and New Developments”, in *Proceedings of 6th IEEE International Conference on Industrial Informatics (INDIN 2008)* in Daejeon, Korea, 13–16 July 2008, IEEE, New York, pp. 569–574.

Campos, J. (2009), “Development in the Application of ICT in Condition Monitoring and Maintenance”, *Computers in Industry*, Vol. 60, No. 1, pp. 1–20.

Chandola, V., Banerjee, A., and Kumar, V. (2009), “Anomaly Detection: A Survey”, *ACM Computing Surveys*, Vol. 41, No. 3, pp. 15:1–15:58.

Chen, S., and Abu-Nimeh, S. (2011), “Lessons from Stuxnet”, *IEEE Computer*, Vol. 44, No. 4, pp. 91–93.

Chesbrough, H. W. (2003), *Open Innovation: The New Imperative for Creating and Profiting from Technology*, Harvard Business School Press, Boston, MA, USA.

Chowdhury, S., and Akram, A. (2011), “e-Maintenance: Opportunities and Challenges”, in *The 34th Information Systems Research Seminar in Scandinavia (IRIS 2011)* in Turku, Finland, 16–19 August 2011, Turku Centre for Computer Science, pp. 68–81.

Chryssolouris, G., Mavrikios, D., Papakostas, N., Mourtzis, D., Michalos, G., and Georgoulias, K. (2009), “Digital Manufacturing: History, Perspectives, and Outlook”, *Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture*, Vol. 223, No. 5, pp. 451–462.

Cocchi, A., and Lazzarini, R. (2010a), *Dispositivo di controllo della carica batterica di un prodotto alimentare liquido o semiliquido*, Italian Patent 0001400195.

Cocchi, A., and Lazzarini, R. (2010b), *Metodo per la determinazione della carica batterica nell'acqua*, Italian Patent 0001402854.

Cocchi, A., and Lazzarini, R. (2011), *Macchina per la produzione di gelato*, Italian Patent 0001405965.

Crespo-Màrquez, A. (2007), *The Maintenance Management Framework*, Springer-Verlag, London, UK.

DaCosta, F. (2013), *Rethinking the Internet of Things: a scalable approach to connecting everything*, Apress, Berkeley, CA, USA.

De Castro, L. N., and Timmis, J. (2002), *Artificial Immune Systems: A New Computational Intelligence Approach*, Springer Science & Business Media.

Di Giacomo, M. (2005), "MySQL: lessons learned on a digital library", *IEEE Software*, Vol. 22, No. 3, pp.10-13.

DiUlio, M., Savage, C., Finley, B., and Schneider, E. (2003), "Taking the Integrated Condition Assessment System to the Year 2010", in *Proceedings of 13th International Ship Control Systems Symposium (SCSS)*, Orlando, FL, USA.

Djurdjanovic, D., Lee, J., and Ni, J. (2003), "Watchdog Agent—an infotronics-based prognostics approach for product performance degradation assessment and prediction", *Advanced Engineering Informatics*, Vol. 17, No. 3-4, pp. 109–125.

Dragomir, O. E., Gouriveau, R., Dragomir, F., Minca, E., and Zerhouni, N. (2009), "Review of Prognostic Problem in Condition-Based Maintenance", in *European Control Conference, ECC'09*, Budapest, Hungary.

Foschini, L., Taleb, T., Corradi, A., and Bottazzi, D. (2011), "M2M-Based Metropolitan Platform for IMS-Enabled Road Traffic Management in IoT", *IEEE Communications Magazine*, Vol. 49, No. 11, pp. 50–57.

Freitas, A., Curry, E., Oliveira, J. G., and O'Riain, S. (2012), "Querying Heterogeneous Datasets on the Linked Data Web – Challenges, Approaches, and Trends", *Internet Computing*, Vol. 16, No. 1, pp. 24–33.

Ganti, R. K., Fan, Y., and Hui, L. (2011), "Mobile Crowdsensing: Current State and Future Challenges", *IEEE Communications Magazine*, Vol. 49, No. 11, pp. 32–39.

Garcia, M. C., Sanz-Bobi, M. A., and del Pico, J. (2006), "SIMAP: Intelligent System for Predictive Maintenance: Application to the Health Condition Monitoring of a Windturbine Gearbox", *Computers in Industry*, Vol. 57, No. 6, pp. 552–568.

Goebel, K., Saha, B., and Saxena, A. (2008), "A Comparison of Three Data-Driven Techniques for Prognostics", in *Proceedings of the 62nd Meeting of the Society For Machinery Failure Prevention Technology (MFPT)*, Virginia Beach, VA, USA, pp. 119-131.

Grossi, M., Lanzoni, M., Lazzarini, R., and Riccò, B. (2012), "Automatic ice-cream characterization by impedance measurements for optimal machine setting", *Measurement*, Vol. 45, No. 7, pp. 1747-1754.

Grossi, M., Lanzoni, M., Pompei, A., Lazzarini, R., Matteuzzi, D., and Riccò, B. (2010), "An embedded portable biosensor system for bacterial concentration detection", *Biosensors and Bioelectronics*, Vol. 26, No. 3, pp. 983-990.

Grossi, M., Lazzarini, R., Lanzoni, M., Pompei, A., Matteuzzi, D., and Riccò, B. (2013), "A Portable Sensor With Disposable Electrodes for Water Bacterial Quality Assessment", *IEEE Sensors Journal*, Vol. 13, No. 5, pp. 1775-1782.

Grossi, M., Lazzarini, R., Lanzoni, M., and Riccò, B. (2011), "A novel technique to control ice cream freezing by electrical characteristics analysis", *Journal of Food Engineering*, Vol. 106, No. 4, pp. 347-354.

Hadden, G.D., Bergstrom, P., Bennett, B.H., Vachtsevanos, G.J., and Van Dyke, J. (2002), "Distributed multi-algorithm diagnostics and prognostics for US Navy ships", in *2002 AAAI Spring Symposium*, Palo Alto, California, CA, USA.

Hedjazi, D., and Zidani, A. (2011), "Development of an Industrial e-Maintenance System Integrating Groupware Techniques", *International Journal of Industrial and Systems Engineering*, Vol. 9, No. 2, pp. 227–247.

Holler, J., Tsiatsis, V., Mulligan, C., Avesand, S., Karnouskos, S., and Boyle, D. (2014), From Machine-to-Machine to the Internet of Things: Introduction to a New Age of Intelligence, Academic Press.

Holmberg, K., Adgar, A., Arnaiz, A., Jantunen, E., Mascolo, J., and Mekid, S. (Eds.) (2010), *e-Maintenance*, Springer, London, UK.

Hung, M.-H., Cheng, F.-T., and Yeh, S.-C. (2003), "Development of a Web-services-based e-Diagnostics Framework", in *Proceedings of the 2003 IEEE International Conference on Robotics and Automation (ICRA 2003)*, Taipei, Taiwan, 14–19 September 2003.

Ishaq, I., Carels, D., Teklemariam, G. K., Hoebeke, J., Abeele, F. V. D., Poorter, E. D., Moerman, I., and Demeester, P. (2013), "IETF Standardization in the Field of the Internet of Things (IoT): A Survey", *Journal of Sensor and Actuator Networks*, Vol. 2, No. 2, pp. 235–287.

Jantunen, E., Emmanouilidis, C., Arnaiz, A., and Gilabert, E. (2011), "e-Maintenance: Trends, Challenges and Opportunities for Modern Industry", in *Proceedings of the 18th IFAC World Congress*, Elsevier, Amsterdam, Holland, pp. 453–458.

Jardine, A., Lin, D., and Banjevic, D. (2006), "A Review on Machinery Diagnostics and Prognostics Implementing Condition-Based Maintenance", *Mechanical Systems and Signal Processing*, Vol. 20, No. 7, pp. 1483–1510.

Johnson P. (2011), "Commercialization of Prognostics Systems Leveraging Commercial Off-The-Shelf Instrumentation, Analysis, and Data Base Technologies", in *Proceedings* *of Annual Conference of the Prognostics and Health Management Society 2011*, Montreal, Quebec, Canada , Vol. 2, pp. 114-122.

Katusic, D., Weber, M., Bojic, I., Jezic, G., and Kusek, M. (2012), “Market, Standardization, and Regulation Development in Machine-to-Machine Communications”, in *Proceedings of 20th International Conference on Software, Telecommunications and Computer Networks (SoftCOM 2012)*, IEEE, New York, NY, USA, pp. 1–7.

Kim, S.-H., Cohen, M., and Netessine, S. (2007), “Performance Contracting in After-Sales Service Supply Chains”, *Management Science*, Vol. 53, No. 12, pp. 1843–1858.

Kumar, S., and Pecht, M. (2010), “Modeling Approaches for Prognostics and Health Management of Electronics”, *International Journal of Performability Engineering*, Vol. 6, No. 5, pp. 467-476.

Kunze, U. (2003), “Condition Telemonitoring and Diagnosis of Power Plants Using Web Technology”, *Progress in Nuclear Energy*, Vol. 43, No. 1–4, pp. 129–136.

Lazzarini, R., and Cocchi, A. (2011), *Macchina per preparazione di gelato*, Italian Patent 0001407015.

Lazzarini, R., Stefanelli, C., Tortonesi, M., and Virgilli, G. (2013), “e-Maintenance for Household and Similar Appliances”, *International Journal of Productivity and Quality Management*, Vol. 12, No. 2, pp. 141-160.

Lazzarini, R., Stefanelli, C., and Tortonesi, M. (2014), “Leveraging ICT to Enable e-Maintenance for Automated Machines”, in *Adaptive, Dynamic, and Resilient Systems*, CRC Press, Taylor & Francis Group, Boca Raton, FL, USA, pp. 215-236.

Lebold, M., and Thurston, M. (2001), “Open Standards for Condition-Based Maintenance and Prognostic Systems”, in *Proceedings of 5th Annual Maintenance and Reliability Conference (MARCON 2001)*, Gatlinburg, USA.

Lee, J., Ni, J., Djurdjanovic, D., Qiu, H., and Liao, H. (2006), “Intelligent Prognostics Tools and e-Maintenance”, *Computers in Industry*, Vol. 57, No. 6, pp. 476–489.

Lee, J., Ghaffari, M., and Elmeligy, S. (2011), "Self-Maintenance and Engineering Immune Systems: Towards Smarter Machines and Manufacturing Systems", *Annual Reviews in Control*, Vol. 35, No. 1, pp. 111-122.

Letuchy, E., (2008), "Facebook Chat", available at: www.new.facebook.com/note.php?note_id= 14218138919&id=9445547199&index= (accessed 7 February 2015).

Levrat, E., and Iung, B. (2007), "TELMA: A full e-Maintenance platform", in *Proceedings of the Second World Congress on Engineering Asset Management and the Fourth International Conference on Condition Monitoring (WCEAM/CM 2007)*, Harrogate, UK, 11–14 June 2007.

Levrat, E., Iung, B., and Crespo-Màrquez, A. (2008), "e-Maintenance: Review and Conceptual Framework", *Production Planning & Control*, Vol. 9, No. 4, pp. 408–429.

Lopez, L. (2007), "Advanced electronic prognostics through system telemetry and pattern recognition methods", *Microelectronics Reliability*, Vol. 47, Issue 12, Pages 1865–1873.

Martin-Flatin, J. P., Jakobson, G., and Lewis, L. (2007), "Event Correlation in Integrated Management: Lessons Learned and Outlook", *Journal of Network and Systems Management*, Vol. 15, No. 4, pp. 481–502.

Medjaher, K., Camci, F., and Zerhouni, N. (2012), "Feature Extraction and Evaluation for Health Assessment and Failure Prognostics", in *First European Conference of the Prognostics and Health Management Society 2012*, Dresden Germany, Vol. 3.

Mitchell, H. B. (2012), *Data Fusion: Concepts and Ideas*, 2nd Edition, Springer, Berlin Heidelberg, Germany.

Mobley, R. K. (2002), *An Introduction to Predictive Maintenance*, 2nd Edition, Butterworth-Heinemann, Boston, MA, USA.

Moubray, J. (1997), *Reliability-Centered Maintenance*, 2nd Edition, Industrial Press Inc., New York, NY, USA.

Muller, A., Crespo-Màrquez, A., and Iung, B. (2008), “On the Concept of e-Maintenance: Review and Current Research”, *Reliability Engineering and System Safety*, Vol. 93, No. 8, pp. 1165–1187.

Nakajima, S. (1988), *Introduction to TPM: Total Productive Maintenance*, Productivity Press, Cambridge, MA, USA.

Oliva, R., and Kallenberg, R. (2003), “Managing the Transition from Products to Services”, *International Journal of Service Industry Management*, Vol. 14, No. 2, pp. 160–172.

Papathanasiou, N., Emmanouilidis, C., Pistofidis, P., and Karampatzakis, D. (2012), "E-Learning and Context Aware e-Support Software for Maintenance", *Journal of Physics: Conference Series*, Vol. 364, No. 1, p. 012110, IOP Publishing.

Pecht, M., and Kumar, S. (2008),“Data Analysis Approach for System Reliability, Diagnostics and Prognostics”, *Pan Pacific Microelectronics Symposium*, Hawaii, USA.

Pfisterer, D., Romer, K., Bimschas, D., Kleine, O., Mietz, R., Truong, C., Hasemann, H., Kröller, A., Pagel, M., Hauswirth, M., Karnstedt, M., Leggieri, M., Passant, A., and Richardson, R. (2011), “SPITFIRE: Toward a Semantic Web of Things”, *IEEE Communications Magazine*, Vol. 49, No. 11, pp. 40–48.

Pistofidis, P., Emmanouilidis, C., Koulamas, C., Karampatzakis, D., Papathanassiou, N. (2012), “A Layered e-Maintenance Architecture Powered by Smart Wireless Monitoring Components”, in *Proceedings of the 2012 IEEE International Conference on Industrial Technology (ICIT 2012)*, Athens, Greece, 19-21 March 2012.

Poongodai, A., and Bhuvaneswari, S. (2013), "AI Technique in Diagnostics and Prognostics", in *Proceedings on National Conference on Future Computing 2013*, International Journal of Computer Applications IJCA, No. 1.

Porter, E. M., and Heppelmann, E. J. (2014), "How Smart, Connected Products Are Transforming Competition", *Harvard Business Review*, Vol. 92, No. 11, pp. 64–88.

Poweski, B., and Raphael, D. (2010), *Security on Rails*, Pragmatic Bookshelf, USA.

Prakash, O. (2006), "Asset Management through Condition Monitoring – How It May Go Wrong: A Case Study", in *Proceedings of 1st World Congress on Engineering Asset Management (WCEAM 2006)*, Springer, New York, NY, USA.

Reshef, D., Reshef, Y., Finucane, H., Grossman, S., McVean, G., Turnbaugh, P., Lander, E., Mitzenmacher, M., and Sabeti, P. (2011), "Detecting Novel Associations in Large Data Sets", *Science*, Vol. 334, No. 6062, pp. 1518–1524.

Ribeiro, L., Barata, J., and Silvério, N. (2008), "A High Level e-Maintenance Architecture to Support On-Site Teams", *Enterprise and Work Innovation Studies*, Vol. 4, IET, pp. 129–138.

Roemer, M. J., Byington, C. S., Kacprzynski, G. J., and Vachtsevanos, G. (2006), "An Overview of Selected Prognostic Technologies with Application to Engine Health Management", in *Proceedings of ASME Turbo Expo 2006 (GT2006) Power for Land, Sea, and Air*, ASME, New York, , NY, USA.

Roemer, M. J., Byington, C. S., Kacprzynski, G. J., Vachtsevanos, G., Goebel, K. (2011), "Prognostics", in *System Health Management: with Aerospace Applications*, John Wiley & Sons, Chichester, UK.

Ruby, S., Thomas, D., and Heinemeier-Hansson, D. (2013), *Agile Web Development with Rails 4*, Pragmatic Bookshelf, USA.

Sakr, S., Liu, A., Batista, D. M., and Alomari, M. (2011), "A Survey of Large Scale Data Management Approaches in Cloud Environments", *IEEE Communications Surveys & Tutorials*, Vol. 13, No. 3, pp. 311–336.

Saxena, A., Celaya, J., Balaban, E., Goebel, K., Saha, B., Saha, S., and Schwabacher, M. (2008), "Metrics for evaluating performance of prognostic techniques", in *International Conference on Prognostics and Health Management 2008 (PHM 2008)*, Denver, CO, USA, 1-17.

Schwabacher, M., and Goebel, K. (2007), "A Survey of Artificial Intelligence for Prognostics", in *2007 AAAI Fall Symposium*, Arlington, VA, USA.

Shamsuzzoha, A. H. M., and Helo, P. T. (2009), "Reconfiguring product development process in auto industries for mass customization", *International Journal of Productivity and Quality Management*, Vol. 4, No. 4, pp. 400–417.

Somayaji, A., Hofmeyr, S., and Forrest, S. (1997), "Principles of a computer immune system", in *Proceedings of the 1997 Workshop on New Security Paradigms (NSPW 1997)*, ACM Press, New York, USA, pp. 75–82.

Sum, C.-S., Harada, H., Kojima, F., Lan, Z., and Funada, R. (2011), "Smart Utility Networks in TV White Space", *IEEE Communications Magazine*, Vol. 49, No. 7, pp. 132–139.

Timmis, J., Hone, H., Stibor, D., and Clark, E. (2008), "Theoretical advances in artificial immune systems", *Theoretical Computer Science*, Vol. 403, No. 1, pp.11–32.

Vandermerwe, S., and Rada, J. (1988), "Servitization of Business: Adding Value by Adding Services", *European Management Journal*, Vol. *6*, No. 4, pp. 314-324.

Vasseur, J. P., and Dunkels, A. (2010), *Interconnecting smart objects with ip: The next internet*, Morgan Kaufmann, Burlington, MA, USA.

Veldman, J., Klingenberg, W., and Wortmann, H. (2011), "Managing Condition-Based Maintenance Technology", *Journal of Quality in Maintenance Engineering*, Vol. 17, No. 1, pp. 40–62.

Vermesan, O., and Friess, P. (Eds.) (2014), *Internet of Things: From Research and Innovation to Market Deployment*, River Publishers , Aalborg, Denmark

Vinoski, S. (2007a), “Concurrency with Erlang”, *IEEE Internet Computing*, Vol. 11, No. 5, pp. 90-93.

Vinoski, S. (2007b), “Reliability with Erlang”, *IEEE Internet Computing*, Vol. 11, No. 6, pp. 79-81.

Voisin, A., Levrat, E., Cocheteux, P., and Iung, B. (2010), “Generic Prognosis Model for Proactive Maintenance Decision Support: Application to Pre-industrial e-Maintenance Test Bed”, *Journal of Intelligent Manufacturing*, Vol. 21, No. 2, pp. 177–193.

Von Hippel E. (2001), “Innovation by user communities: Learning from open source software”, *MIT Sloan Management Review*, Vol. 42, No. 4, pp. 82–86.

Vosloo, I., and Kourie, D. (2008), “Server-Centric Web Frameworks: An Overview”, *ACM Computing Surveys*, Vol. 40, No. 2, Article 4.

Wan Mahmood, W. H., Ab Rahman, M. N. A., Deros, B. M., and Mazli, H. (2011), “Maintenance Management System for Upstream Operations in Oil and Gas Industry: A Case Study”, *International Journal of Industrial and Systems Engineering*, Vol. 9, No. 3, pp. 317–329.

Wang, H., Song, B., and Wan, F. (2011), “Research on fault pattern recognition for aircraft fuel system with its performance simulation”, in *2011 2nd International Conference on Artificial Intelligence, Management Science and Electronic Commerce (AIMSEC)*, Deng Leng , pp. 4231-4235.

Wu, G., Talwar, S., Johnsson, K., Himayat, N., and Johnson, K. (2011), “M2M: From Mobile to Embedded Internet”, *IEEE Communications Magazine*, Vol. 49, No. 4, pp. 36–43.

Yang, Q., Barria, J. A., and Green, T. C. (2011), “Communication Infrastructures for Distributed Control of Power Distribution Networks”, *IEEE Transactions on Industrial Informatics*, Vol. 7, No. 2, pp. 316–327.

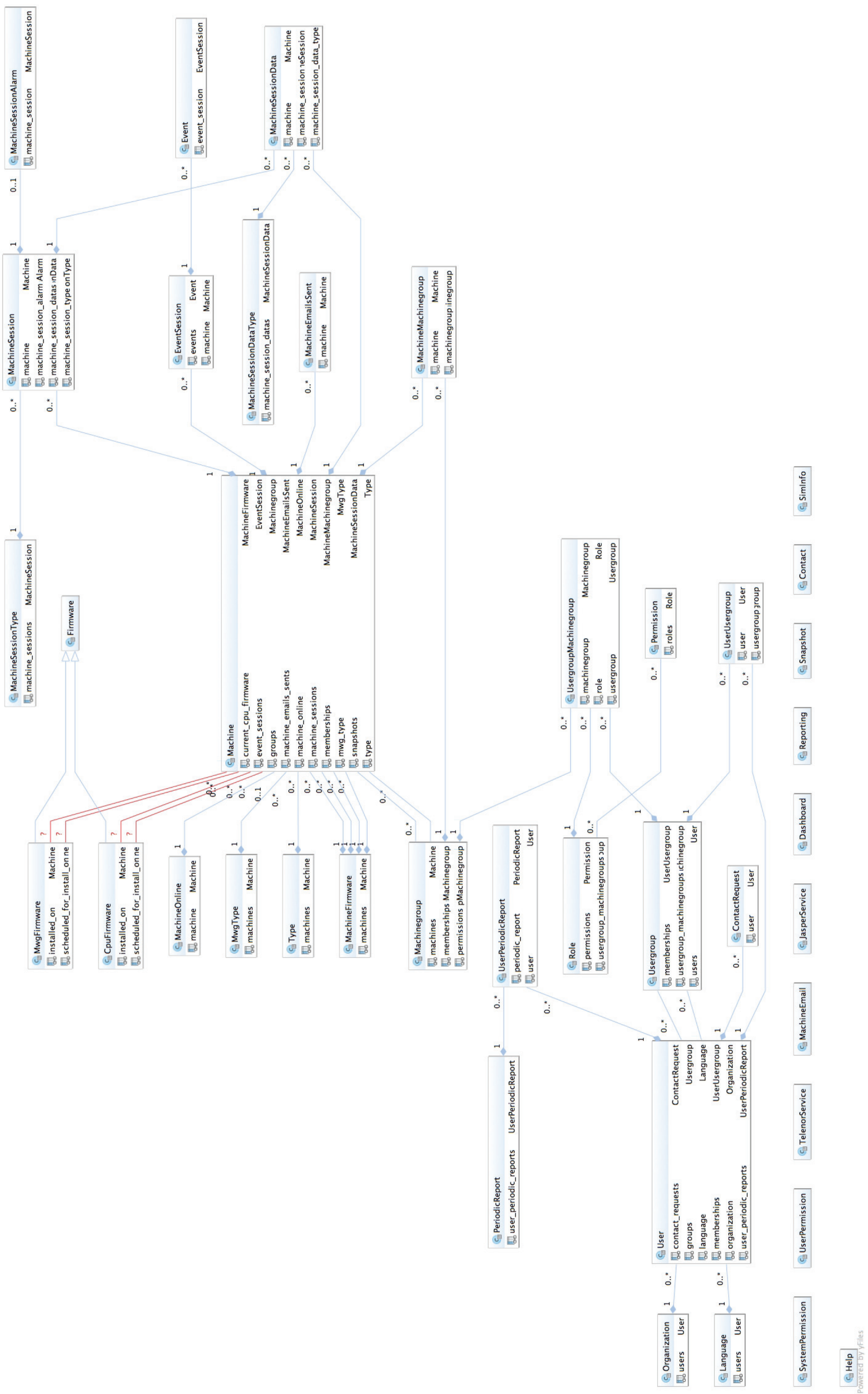
Zephir, O., and Minel, S. (2007), "Reaching readiness in technological change through the application of capability maturity models principals”, in *Complex Systems Concurrent Engineering*, Springer, London, UK, pp. 57-64.

Zhou, H. (2012), *The Internet of Things in the Cloud: A Middleware Perspective*, CRC Press, Taylor & Francis Group, Boca Raton, FL, USA.

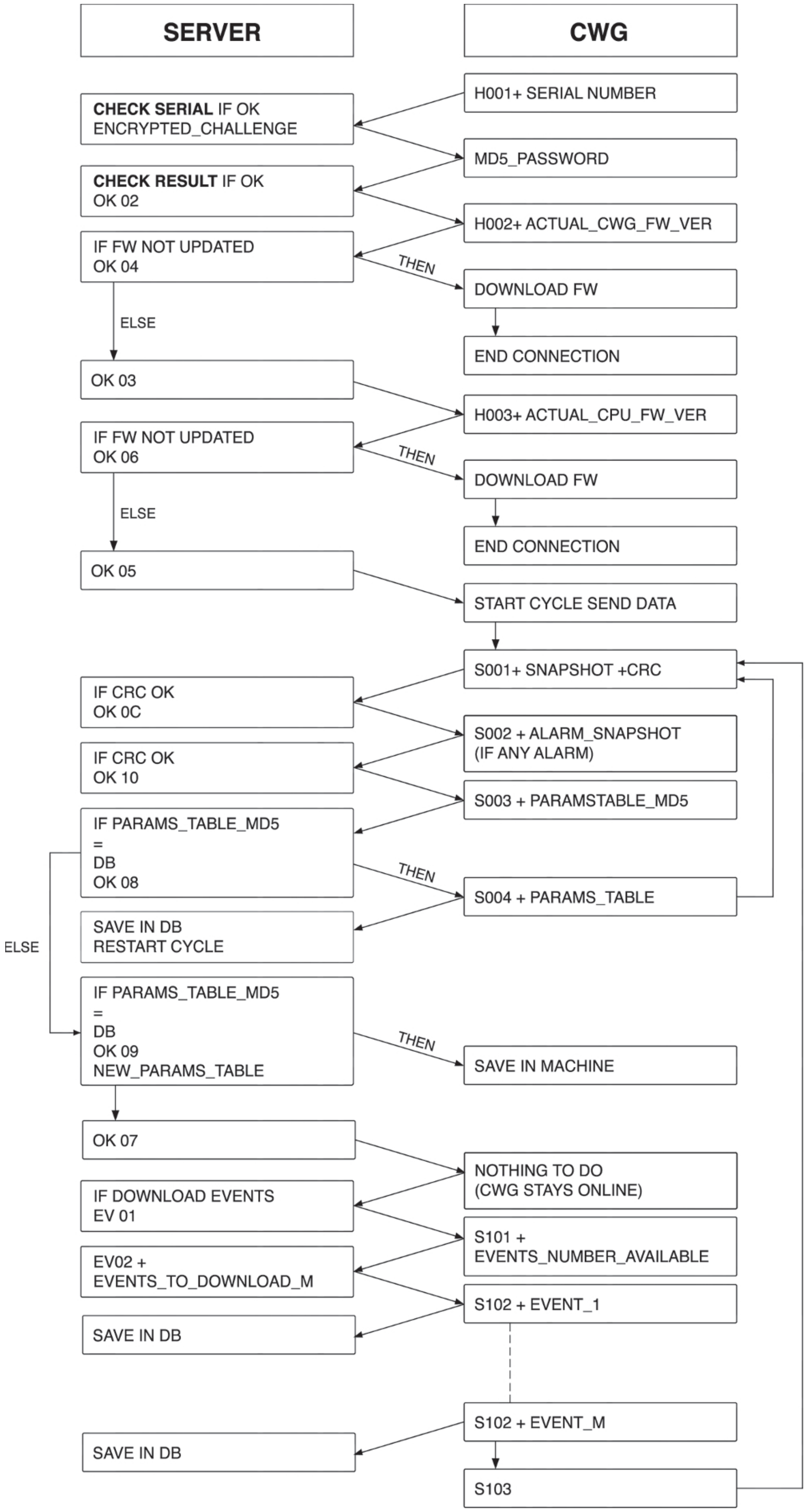
Zhu, J. (2003), “Web Services Provide the Power to Integrate”, *IEEE Power and Energy Magazine*, Vol. 1, No. 6, pp. 40–49.

Appendices

Appendix A
The Teorema Database



Appendix B
The Teorema Protocol



Appendix C

The Teorema Daemon

```
#!/usr/bin/perl
#
# TEOREMADAEMON - Changelog
#
# 0.1.0-0.3.0 Try again Sam
# 0.4.0 simple authentication with serial numbers
# 0.5.0 bugs fixed and improvements (goDaemon)
# 0.6.0  Added authentication challenge and firmware check
#        Run as user
# 0.6.1 Check boolean flag_update in database to enable firmware upgrade
#        Show version
# 0.6.2 Check distinct flag_update_fw_mwg flag_update_fw_cpu
# 0.6.3 Better log
# 0.6.4 ID_DEV->DEV_SERIALNUM
#        Parameters management
#        Crop snapshot in log
# 0.6.5 different behaviour when mwgate send current params/MD5
# 0.6.6 added link to mail script for sending mails at alarms
# 0.6.7 added the insert of first regular snapshot into storico table
#        flag_update_fw_cpu setted to ZERO at the end of connection, if previously setted to ONE
# 0.6.8 changed $daemon_uid e $daemon_gid from 48 to 33
# 0.7.0 fixed multiple DBI connection: now the connection from the child to the DB with an appropriate
$child_dbh - create by the clone func - and not using the parent $dbh
#        fixed the sub for handling the sig_alarm for the tcp timeout
#        removed unused sub for type conversions (are commented)
#        fixed the writing into the log of the "unknown event", previously thise line was write at each
loops cycle (the logsize in a session became about 300MB)
# 0.7.1 add the "OK 10" response when an alarm snapshot is received. In case of alarm, the mwg send the
alarm snapshot and close the connection: there is no duration connection of 2 minutes
#                during an alarm connection in order to reduce the traffic on air (and the money
spent). The string OK 10 is sent only whit the firmware version >= 1.03.
#        improoved the logs when a mwg gives "Auth Wrong" with an indication of the challenge/passwd
sent
# 0.7.2 fix a bug when a techinichian call the mwg. The only snapshot inserted was the alarm type.
Insert the variable $unknown_err_count intead of using $insert for test the unknown error.
# 1.0.0 update the query with the new database schema of the Teorema project.
# 1.0.1 Updated the URL of the GET for emails
# 1.0.2-TEST Changed TCP Port (to 8898) for application test connections. Insert query's for set/update
the online/offline/connecting flag's
# 1.0.3-TEST Added into log the versions of mwg and cpu firmwares.
#            Sostituito localhost con URL teorema.macchinepergelato.it al richiamo dello script per
l'invio delle email, cambiato url
# 1.0.4-TEST Added query that updates the machine_onlines flags when connection drops if the auth is
wrong.
# 1.0.5 Changed database
# 1.0.6 Changed mysql command NOW() to UTC_TIMESTAMP(
#
# TEOREMA/DIALOG DAEMONS - Changelog
#
# 1.0    Added commands OK11 (mwg goes offline) after the mwg command S003, which ask for the parameters
table
#        Added "if" that permits to modify the machine_session_type if the snapshot is an alarm snapshot
(S002) after few 'normal' (S001) snapshots
#        Changed daemon name and strings to be written on the logfile
# 1.0.1 Check if the alarm is secondary before recording it in the DBMS.
# 1.0.2 Added pasteurization alarms
#        BUGFIX: added the next_parameters update when receiving the current_parameters (S004)
# 1.0.3 Added events download
#        Added "Remote machine block" command to be sent to the machine (H005, OK 11 and OK 12)
# 1.0.4 Added management of webgates auto connection (wg sends S005 after the login to tell the Daemon
that the current session has been automatically generated.
#        When the Daemon finds S005 sends the normal events request to wg.
#        Added the management of both daemons Teorema/Dialog into logs and filenames
#        Refactored code
# 1.0.5 Added the machines.last_snapshot_id and the machine_session_datas.machine_id updates when
receiving a new snapshot
# 1.0.6 @pasteurization_alarms: removed "Mancata Pasto" and "Pasto not done" alarms; added pasto alarms
in Spanish
#        @secondary_alarms: added Falta Cono!, Cono Equivocado! and Wrong Cone !
#        @pasteurization_alarms: added End Pasto
# 1.0.7 Set machines.events_to_download = 0 if the connection is dropped before the events download has
finished
#        Set machines.flag_update_parameters = false when the connection is dropped
#        Added the update of events.machine_id field when a new event is stored in the db
#        Added filter on machines.disabled flag: select and update only machines with disabled = 0
#        @secondary_alarms: added No Cone!
# 1.0.8  MD5 calculation: since from fw version 1.60 the webgate sends to the concentrator the entire
content of the buffer, we need to detect the end of line character in the
#        string received in order to calculate MD5 on the correct string. In order to do that everytime
we will receive the parameters table we will check the fw version of the
#        webgate. For webgates with fw versions greater than or equal to 1.60 we will perform the
research of the string terminator and we will calculate the MD5 on the correct
```

```
#       string.
# 1.0.9 Webgate Login: if the webgate sends as serial number 00000000 we need to check the ip address
of the sim in it installed in order to authenticate it.
# 1.1.0 Download Events: the flag_download_events will be set to false only after an event session or
after 15 minutes since the click of the button "Start Download Events".
#       Before this change it was set to false after sending EV 02 to the webgate. In this way if the
events download wasn't able to start during the current session the user
#       was not able to download them anymore.
# 1.1.1 Added "OK 13 NumFile" command sent to webgate after receiving H006 from device. Introduced to
tell to the webgate how many files must be downloaded for firmware upgrade.
#       Used ONLY for u-gate devices (UMTS).
# 1.1.2 Added "OK 0C", "OK 0D" and "OK 0E" for sending to webgates the request for S001, S002 and S003.
Used with cgate200.
#       Added firmware cgate210 and cgate211
# 1.1.3 Added secondary alarms Alarm IMS1 and Alarm IMS2 for PastomasterRTL.
# 1.1.4 Added primary and secondary alarms to the current list.
# 1.1.5 Updated alarms list
# 1.1.6 Bugfix on automatic download events.
# 1.1.7 Added serials 00100001 and 10100001 in the list of serial numbers for which it's looked the ip
address.
#

my $version="1.1.7";
$| = 1;

########################################################################## TEOREMA INIT STRING
print "$0 [$$]: TEOREMA DAEMON - Version $version - Initializing....\n";
########################################################################## DIALOG INIT STRING
#print "$0 [$$]: DIALOG DAEMON - Version $version - Initializing....\n";

use strict;
use vars ('$PLCSOCK','$httpsocket');
use POSIX qw(setsid strftime);
use DBI;
use IO::Socket;
use IO::Select;
use Digest::MD5;
use DateTime;
use DateTime::Format::Strptime;

# Module not installed by default
use Proc::UID qw(:funcs);
#####USER AGENT#####
require LWP::UserAgent;

my $waitedpid = 0;

#CONFIGURATION VARIABLES

#MySQL
my $DBServer='localhost';
my $DBUser='production_user';
my $DBPassword='mycdcpro';
############################## TEOREMA TEST DB
#my $DBName='cdc_development';
############################## TEOREMA DB
my $DBName='cdc_production';
############################## DIALOG DB
#my $DBName='dialog_production';

#User, port, timeout, files
my $daemon_uid=33;
my $daemon_gid=33;
############################## TEOREMA TEST PORT
#my $tcp_port=8888;
############################## TEOREMA PORT
my $tcp_port=8898;
############################## DIALOG PORT
#my $tcp_port=8998;

my $maxtries=15;
my $tcp_timeout=60;                                #timeout waiting for data (GPRS connection dropped)

##################################################### TEOREMADAEMON TEST PID & LOG FILES
#my $pidfile     = "/var/run/teorema-daemon-test.pid";
#my $logfile     = "/var/www/cdc/teorema-daemon-test.log";
##################################################### TEOREMADAEMON PID & LOG FILES
my $pidfile      = "/var/run/teorema-daemon.pid";
my $logfile      = "/var/log/teorema-daemon.log";
##################################################### DIALOGDAEMON PID & LOG FILES
#my $pidfile     = "/var/run/dialog-daemon.pid";
#my $logfile     = "/var/log/dialog-daemon.log";

##### USER AGENT #####
my $ua = LWP::UserAgent->new;

############################################## TEOREMA LOG STRING
$0 = 'teoremadaemon [master:'. $tcp_port.']';
############################################## DIALOG LOG STRING
#$0 = 'dialogdaemon [master:'. $tcp_port.']';
```

```
# Do the daemon thang:
sub goDaemon
{
	chdir '/' or die "Cannot chdir to / ($!)";

	open STDIN,  "/dev/null"   or die "Cannot read /dev/null ($!)";
	open STDOUT, ">> $logfile" or die "Cannot write to $logfile ($!)";
	open STDERR, ">> $logfile" or die "Cannot write to $logfile ($!)";

	defined(my $pid = fork) or die "Cannot fork ($!)";
	exit if $pid;
	setsid or die "Cannot start a new session ($!)";
	umask 0;

	# Write pid file
	open PIDFILE, "> $pidfile" or die "Cannot write to $pidfile ($!)\n";
	print PIDFILE "$$\n";
	close PIDFILE;

	# Open log file
	open LOGFILE, ">> $logfile" or die "Cannot write to $logfile ($!)\n";

	# Drop privileges
	drop_gid_perm($daemon_gid);
	drop_uid_perm($daemon_uid);
}

### LOGMSG BEGIN ###
# Writes on the log file
sub logmsg
{
	print STDOUT "$0 [$$]: ", scalar localtime, " : @_\n"
}
### LOGMSG END ###

### CREATESESSION BEGIN ###
# Creates a new session to store the snapshot on the DB.
#	USAGE:
#		&CreateSession($session_type, $data_type, $machine_id)
#	VALUES:
#		$session_type	=>	"alarm" | "monitor"
#		$data_type	=>	"snapshot"
#		$machine_id	=>	machine_id
sub CreateSession
{
	my $sid;				# return value
	our $machine_session_type_id;		# defined in the main
	our $machine_session_data_type_id;# defined in the main
	our $child_dbh;				# defined in the main

	# Finds infos to create the session
	my $res = $child_dbh->selectall_arrayref("SELECT * FROM machine_session_types WHERE name = ?",
undef, $_[0])	# $_[0] is the session type. Must be "alarm" or "monitor"
		or die logmsg $child_dbh->errstr;
	$machine_session_type_id = $$res[0][0];		#the id is at position 0

	$res = $child_dbh->selectall_arrayref("SELECT * FROM machine_session_data_types WHERE name
= ?", undef, $_[1])	# $_[1] is the data type
		or die logmsg $child_dbh->errstr;
	$machine_session_data_type_id = $$res[0][0];	#the id is at position 0

	# Creates the session
	$child_dbh->do("INSERT INTO machine_sessions (machine_session_type_id, timestamp_begin,
timestamp_end, machine_id) VALUES (?, UTC_TIMESTAMP(), UTC_TIMESTAMP(), ?) ", undef,
$machine_session_type_id, $_[2])  # $_[2] is the machine id
		or die logmsg $child_dbh->errstr;

	# Searches for the session ID of the session just created
	$res = $child_dbh->selectall_arrayref("SELECT * FROM machine_sessions WHERE machine_id = ?
ORDER BY timestamp_begin DESC LIMIT 1", undef, $_[2])	# $_[2] is the machine_id
		or die logmsg $child_dbh->errstr;
	$sid = $$res[0][0];	#the id is at position 0 0

	return $sid;
}
### CREATESESSION END ###

### CREATESESSIONALARM BEGIN ###
# Creates a new session to store the snapshot on the DB
#	USAGE:
#		&CreateSessionAlarm($machine_session_id, $alarm_name)
sub CreateSessionAlarm
{
	my $sid;		# return value
	our $child_dbh;		# global variable defined in the main

	# $_[0] is the machine_session_id and $_[1] is the alarm_name
	my $machine_session_id = $_[0];
	my $alarm_name = $_[1];
```

```
        # List of alarms that don't block the machine
        my @secondary_alarms = ("Spigot Opened   ","  Portello  Aper","Alarm IMS       ","Tapkop Open
","A09             ","Power On        ",
                        "Portello Aperto ","Alarm 09 IMS    "," No more cones! "," Lavare oggi!   ","
IMSA Aperto    "," IMSB Aperto    ",
                        " IMSA Open      "," IMSB Open      "," IMSA Ouvert    "," IMSB Ouvert    ","
IMSA Offen     "," IMSB Offen     ",
                        "   Spigot  Opene"," Fronttur offen "," Wash today!    "," Heute waschen!
","Allarme IMS     ","Laver ce jour!  ",
                        "Prtillon Absent ","    Cover Aperta","Reinig Vandaag! ","  Cono Errato!  ","
Keine Waffel ! "," Waffel Falsch! ",
                        " Cono Mancante! ","  Uitloop Open  ","  Wrong Cone !  ","    No Cone !   ","
Falta Cono!  ","Cono Equivocado!",
                        "Alarm IMS1      ","Alarm IMS2      ","Beschermkap Open","    Cover
Aperta","Allarme IMS2    ","Mancata Tensione",
                        "Retorno Tension "," Invert Phases! "," Invertire Fasi "," Invertir  Fasi
","Inverser Phases!","InverseazaFazele",
                        "Inverter Fases  ","   Fasefeil!    "," Spigot Closed  ","Table Updated
M.","Table Updated R.","Aggiorn.Tab.Macc",
                        "Aggiorn.Tab.Rem.","Param.oppdatertM","Pasteurisieren! "," Pastorizzare !
","Pasteuriseer nu!"," Pasteuriser!   ",
                        "  Pasteurizar!  ","Mix Out         ","    Mix Out     "," Mix  Esaurita  ","
Mix  Agotado  ","     Mix OK     ",
                        "Add Mix to Pasto","  Cooling Mix   "," Pasto needed!  "," Coni esauriti!
","Conos Agotados! ","Nao mais cones  ",
                        " No more Cups!  "," No more cones 1"," No more cones 2","Perche' in STOP?","
Max Level    ","Acquisizione ...",
                        "Erwerb Daten ...","Data Acquiring..","Acquisizione Dat","Setting in
corso","Sgocciolio Manca","Cornet Termine  ",
                        "Portillo Abierto","   Lavar hoy!   "," Lavar hoje!    ","Production      ","
Washing Done  ","Washing Not Done",
                        "Mancato Lavaggio","Lavaggio Eseguit","Errore Comun.MB ","Errore Comunic.
","Errore Comunic.P","Err.Comunicazion",
                        "Communic.Error P","Communic.Error E","Errore Comunic.E","Setting in
progr","Verificar conex.","Parameter Chek  ",
                        "Setting in prog."," Autosetup OK   "," CheckSum Fail  ","Keyboard
Blocked","Temporary Mode  ","Close R Handle  ",
                        "Close Mid Handle","Close L Handle  "," Close R Handle "," Close L Handle
","Tensiune Pornita","Reset Hours     ",
                        "Reset Ore Motori","Allarme IMS1    ","Cornet Termine 1","Lipsa cornete 1
","Mauvais Cornet !"," Manque Cornet! ");

        # List of alarms related to the pasteurization cycle
        my @pasteurization_alarms = ("                "," - Ende Pasto - ","-Fine Pastoriz.-","  -Pasto
End-   "," - Pasto End -  ","- Fin de Pasto -",
                           "Inizio Pastoriz.","Inizio Pausa    ","Fine Pausa      ","Start Pasto
","Fin de Pause    ","Pause Ende      ",
                           "Pause           ","Start Pasteur.  ","End Pause       ","Start Pause
","Mancata Pasto   "," Pasto not Done ",
                           " Manque Pasteur.","Past.NichtErfolg","Inicio Pasteur. ","Fin de la
Pasteu","Inicio Pausa    ","Fin Pausa       ",
                           "Pasto End       ");

        # find whether the alarm is in the "secondary" or "pasteurization" list
        my $alarm_type = 0;

        if (grep { $_ eq $alarm_name } @secondary_alarms) {
                $alarm_type = 2;
        }
        elsif (grep { $_ eq $alarm_name } @pasteurization_alarms) {
                $alarm_type = 1;
        }

        # Creates the session that stores infos about the alarm
        $child_dbh->do("INSERT INTO machine_session_alarms (machine_session_id, alarm_name,
alarm_status, created_at, alarm_type) VALUES (?, ?, 1, UTC_TIMESTAMP(), ?)", undef,
$machine_session_id, $alarm_name, $alarm_type)
                or die logmsg $child_dbh->errstr;

        # Searches for the session ID of the session alarm just created
        my $res = $child_dbh->selectall_arrayref("SELECT * FROM machine_session_alarms WHERE
machine_session_id = ? ORDER BY created_at DESC LIMIT 1", undef, $_[0])      # $_[0] is the
machine_session_id
                or die logmsg $child_dbh->errstr;
        $sid = $$res[0][0];       #the id is at position 0 0

        return $sid;     # return the machine_session_alarm_id
}
### CREATESESSIONALARM END ###

### UPDATE LASTSNAPSHOTID BEGIN ###
# Updates last_snapshot_id on the table machines when receiving a new snapshot
#       USAGE:
#               &UpdateLastSnapshotId($machine_id)
sub UpdateLastSnapshotId
{
        our $child_dbh;           # global variable defined in the main

        # $_[0] is the machine_id
```

```
        my $machine_id = $_[0];

        my $res = $child_dbh->selectall_arrayref("SELECT id FROM machine_session_datas WHERE machine_id
= ? ORDER BY id DESC LIMIT 1", undef, $machine_id)
                or die logmsg $child_dbh->errstr;

        $child_dbh->do("UPDATE machines SET last_snapshot_id=? WHERE id=?", undef, $$res[0][0],
$machine_id)
                or die logmsg $child_dbh->errstr;
}
### UPDATE LASTSNAPSHOTID END ###

### CREATEEVENTSESSION BEGIN ###
# Creates a new session on the table event_sessions
#       USAGE:
#               &CreateEventSession($machine_id)
sub CreateEventSession
{
        my $esid;                # return value
        our $child_dbh;          # global variable defined in the main

        # $_[0] is the machine_id
        my $machine_id = $_[0];

        # Creates the event session
        $child_dbh->do("INSERT INTO event_sessions (machine_id, created_at, updated_at) VALUES (?,
UTC_TIMESTAMP(), UTC_TIMESTAMP())", undef, $machine_id)
                or die logmsg $child_dbh->errstr;

        # Searches for the session ID of the session just created
        my $res = $child_dbh->selectall_arrayref("SELECT * FROM event_sessions WHERE machine_id=? ORDER
BY created_at DESC LIMIT 1", undef, $machine_id)
                or die logmsg $child_dbh->errstr;
        $esid = $$res[0][0];     #the id is at the position 0 0

        # Returns the event_session_id
        return $esid;
}
### CREATEEVENTSESSION END ###

### REAPER BEGIN ###
# Creates the signal handler's and associates each handler to his signal
sub REAPER
{
        $waitedpid = wait;
        $SIG{CHLD} = \&REAPER;
        logmsg "reaped child $waitedpid" ;
}
### REAPER END ###

### SIGNALHANDLER BEGIN ###
sub signal_handler
{
        my $signame = shift;
        logmsg "Caught signal $signame, exiting.";
}
### SIGNALHANDLER END ###

### SIGALARMHANDLER BEGIN ###
# Used for the TCP timeout, goes out from the child if called
sub sig_alarm_handler
{
        our $PLCSOCK;
        our $httpsocket;

        our $machine_id;         # defined in the main
        our $machine_session_id; # defined in the main
        our $child_dbh;                  # defined in the main

        # Closes the session only if opened
        if ($machine_session_id != '')
        {
                $child_dbh->do("UPDATE machine_sessions SET timestamp_end=UTC_TIMESTAMP() WHERE id
= ?", undef, $machine_session_id)
                        or die logmsg $child_dbh->errstr;
        }

        # Updates the OFFLINE flag
        $child_dbh->do("UPDATE machine_onlines SET last_session_id=?, offline=true, connecting=false,
connected=false, online=false WHERE machine_id = ?", undef, $machine_session_id, $machine_id)
                or die logmsg $child_dbh->errstr;

        logmsg "Timeout in TCP communication (connection lost?)";
        if( -S $PLCSOCK )
        {
                logmsg "Unlinking socket ".$PLCSOCK;
                $httpsocket->close;
                unlink $PLCSOCK;
        }
        exit(0);
```

```
} ;
### SIGALARMHANDLER END ###

$SIG{CHLD} = \&REAPER;
$SIG{INT} = $SIG{TERM} = $SIG{HUP} = \&signal_handler;
$SIG{ALRM} = \&sig_alarm_handler;

# Flush the buffer
$| = 1;

# Send myself to the background:
&goDaemon;

# System variables
my $tcpsocket;
my $peerhost;
my $row;
my $dbh = DBI->connect("DBI:mysql:database=$DBName:server=$DBServer", $DBUser, $DBPassword)
   or die "Can\'t connect to DB";

# Init query
my $stm= $dbh->prepare("SELECT mwg_serial FROM machines")
        or die "Can\'t prepare query 0-init statement";;
eval { $stm->execute or die $dbh->errstr; };


# Creates the SOCKET on /tmp
while (($row)=$stm->fetchrow_array)
{
        if ( -S '/tmp/PLCSOCK.'.$row )
        {
                unlink '/tmp/PLCSOCK.'.$row;
        }
}

#Creates the listener
my $listener = IO::Socket::INET->new( LocalPort     => $tcp_port,
                                      Listen        => 100,
                                      Reuse         => 1 ) or die "listen: $!";

# Writes the logfile when the daemon starts
################################################################################## TEOREMA LOG STRING
logmsg "TEOREMA DAEMON - Version $version - Init complete. Waiting for connection.";
################################################################################## DIALOG LOG STRING
#logmsg "DIALOG DAEMON - Version $version - Init complete. Waiting for connection.";

### PARENT FOR CYCLE BEGIN ###
for (    $waitedpid = 0; ($tcpsocket = $listener->accept) || $waitedpid;     $waitedpid = 0 )
{
        next if $waitedpid and not $tcpsocket;
        my $pid = fork;
        die "Fork error:  $!" unless defined $pid;

########### FORK BEGIN ###
        if ($pid == 0)
        {
                $peerhost = $tcpsocket->peerhost;

                ################################################################## TEOREMA LOG STRING
                $0 = 'teoremadaemon [client '.$peerhost.']';
                ################################################################## DIALOG LOG STRING
                #$0 = 'dialogdaemon [client '.$peerhost.']';

                logmsg "[".$peerhost."] socket opened";
                # in child here, close unused handler
                $listener->close();
                $tcpsocket->autoflush(1);

                # EVENTUALLY I CAN PUT THIS INSTRUCTION WITH THE OK() FUNC, IN ORDER TO TEST IT
                our $child_dbh = $dbh->clone();            # Global variable

                $dbh->{InactiveDestroy} = 1;      # This tell DBI that the parent's connection should
not to be closed when $dbh is destroyed
                undef $dbh;                       # it's the same to assign the NULL value in C

                # Creates the challenge number, a 8 hex random char, then sends it to the webgate
                my $challenge = substr(Digest::MD5::md5_hex(rand(1000)),0,8);
                print $tcpsocket "OK 01 " .$challenge. "\r\n";

                my $mwg_serial;                   # connected device's serial number
                my @row;

                my @result;                               # array for general purpose query
                my @result_conn;
                my $id;                                   # id for general purpose query
                my $count = $maxtries;                    # number of tries left in authentication
phase

                our $machine_id;                  # global variable. Used in the sig_alarm_handler
```

```
		my $crypted_password = '';			# the right authent answer
		my $crypted_challenge= '';			# the answer sent by connected device

		my $firmware_mwg;			# device webgate fw
		my $firmware_cpu;			# device cpu fw
		my $next_fw_mwg_id;				# the ID of the next webgate fw
		my $next_fw_cpu_id;				# the ID of the next cpu fw
		my $nextfirmware_mwg = '';			# name of the next webgate fw
		my $nextfirmware_cpu = '';			# name of the next cpu fw
		my $flag_update_fw_mwg;			# if TRUE this device should update webgate fw
		my $flag_update_fw_cpu;			# if TRUE this device should update cpu fw

		my $current_parameters;				# device parameters
		my $current_parameters_hash;			# device parameters hash
		my $current_parameters_timestamp;	# device parameters timestamp
		my $next_parameters;				# next device parameters
		my $next_parameters_hash;		# next device parameters hash
		my $next_parameters_timestamp;			# next device parameters timestamp

		my $go_offline;					# go offline flag

		our $machine_session_type_id;			# global variable
		our $machine_session_data_type_id;# global variable
		our $machine_session_id = '';			# global variable. If ( $machine_session_id
!= '') --> don't call the CreateSession sub
		my $machine_session_alarm_id = '';# If ( $machine_session_alarm_id != '') --> don't
call the CreateSessionAlarm sub

		my $last_available_events;			# number of available events sent by the
webgate
		my $events;					# contains the event
		my $event_session_id = '';			# global variable for the events download
sessions

		my $auto;					# flag of the automatic connection
		my $fw_mwg;

		my $event_session_started = 0;
		my $last_snapshot_timestamp;
		my $events_flag_at_session_beginning;

		my $s001_requests = 0;				# stores ho many s001 have been requested

		# Prepares query ID1 that selects the password and fw data
		$stm = $child_dbh->prepare("SELECT crypted_password, next_fw_mwg_id, next_fw_cpu_id,
flag_update_fw_mwg, flag_update_fw_cpu, id FROM machines WHERE mwg_serial = ?  AND disabled = 0")
			or die "Can\'t prepare query 1 statement";

		# Prepares query IDn that selects the password and fw data
		my $stm_ip = $child_dbh->prepare("SELECT crypted_password, next_fw_mwg_id,
next_fw_cpu_id, flag_update_fw_mwg, flag_update_fw_cpu, id FROM machines WHERE ip_address = ?  AND
disabled = 0")
			or die "Can\'t prepare query 1 statement";

################### AUTHENTICATION PROCESS BEGIN ###
		alarm($tcp_timeout);
		do
		{
			$row=0;
			if ($count == 0)
			{
				print $tcpsocket "ERR 03 max num tries reached. BYE.\r\n";
				logmsg "[".$peerhost."] Max num (".$maxtries.") tries reached";
				exit(0);
			}

			if (defined ($mwg_serial = <$tcpsocket>))
			{
				alarm(0);
################################### H001 BEGIN ###
				if( $mwg_serial =~ s/^H001\s+(.*)/$1/)
				{
					$mwg_serial =~ s/\r\n//;
					($mwg_serial,$crypted_challenge)=split /\s+/, $mwg_serial;
					logmsg $mwg_serial.": cryptchallenge [".$peerhost."]";
					if (($mwg_serial ne "00000000") && ($mwg_serial ne
"gate.mac") && ($mwg_serial ne "00100058") && ($mwg_serial ne "00100001") && ($mwg_serial ne
"10100001"))
					{
						# Executes query ID1 (see row 405)
						eval { $stm->execute($mwg_serial); };
						if ($@)
						{
							print $tcpsocket "ERR 01 Server
unavailable.\r\n";
							logmsg $child_dbh->errstr;
							exit(0);
						}
						if (@row=$stm->fetchrow_array)
						{
```

```
        $crypted_password=Digest::MD5::md5_hex($challenge.$row[0]);
                                                            $machine_id = $row[5];          #NB: sono
gli ID!!!
                                                            logmsg "DataBase machine_id: ".
$machine_id."  ";

                                                            if ($crypted_challenge eq $crypted_password
)
                                                            {
                                                                    $next_fw_mwg_id = $row[1];
        #NB: sono gli ID!!!
                                                                    $next_fw_cpu_id = $row[2];
        #NB: sono gli ID!!!
                                                                    $flag_update_fw_mwg = $row[3];
                                                                    $flag_update_fw_cpu = $row[4];

                                                                    # Query that finds the version of
the webgate fw
                                                                    my $stm3 = $child_dbh-
>prepare("SELECT * FROM machine_firmwares WHERE id = ?")
                                                                            or die "Can\'t prepare
query 2 statement";
                                                                    eval { $stm3-
>execute($next_fw_mwg_id); };
                                                                    if ($@)
                                                                    {
                                                                            print $tcpsocket "ERR 01
Server unavailable.\r\n";
                                                                            logmsg $child_dbh->errstr;
                                                                            exit(0);
                                                                    }
                                                                    if (@result=$stm3->fetchrow_array)
                                                                    {
        $nextfirmware_mwg=$result[3];     #the version of the fw is placed at the 3rd position
(starting count from 0)
                                                                    }

                                                                    # Query that finds the version of
the cpu fw
                                                                    eval { $stm3-
>execute($next_fw_cpu_id); };
                                                                    if ($@)
                                                                    {
                                                                            print $tcpsocket "ERR 01
Server unavailable.\r\n";
                                                                            logmsg $child_dbh->errstr;
                                                                            exit(0);
                                                                    }
                                                                    if (@result=$stm3->fetchrow_array)
                                                                    {
        $nextfirmware_cpu=$result[3];     #the version of the fw is placed at the 3rd position
(starting count from 0)
                                                                    }
                                                                    $stm3->finish();
                                                                    logmsg $mwg_serial.": auth OK";
                                                            }
                                                            else
                                                            {
                                                                    logmsg $mwg_serial.": auth wrong
( md5(password+chall) transmitted: ".$crypted_challenge." , challenge sent by the server: ".
$challenge." )";
                                                            }
                                                    }
                                            }else{
                                                    # Executes query IDn (see row 405)
                                                    eval { $stm_ip->execute($peerhost); };
                                                    if ($@)
                                                    {
                                                            print $tcpsocket "ERR 01 Server
unavailable.\r\n";
                                                            logmsg $child_dbh->errstr;
                                                            exit(0);
                                                    }
                                                    if (@row=$stm_ip->fetchrow_array)
                                                    {
        $crypted_password=Digest::MD5::md5_hex($challenge.$row[0]);
                                                            $machine_id = $row[5];          #NB: sono
gli ID!!!
                                                            logmsg "DataBase machine_id: ".
$machine_id."  ";

                                                            $next_fw_mwg_id = $row[1];      #NB: sono
gli ID!!!
```

```
                                                            $next_fw_cpu_id = $row[2];        #NB: sono
gli ID!!!
                                                            $flag_update_fw_mwg = $row[3];
                                                            $flag_update_fw_cpu = $row[4];

                                                            # Query that finds the version of the
webgate fw
                                                            my $stm3 = $child_dbh->prepare("SELECT *
FROM machine_firmwares WHERE id = ?")
                                                                    or die "Can\'t prepare query 2
statement";
                                                            eval { $stm3->execute($next_fw_mwg_id); };
                                                            if ($@)
                                                            {
                                                                    print $tcpsocket "ERR 01 Server
unavailable.\r\n";
                                                                    logmsg $child_dbh->errstr;
                                                                    exit(0);
                                                            }
                                                            if (@result=$stm3->fetchrow_array)
                                                            {
                                                                    $nextfirmware_mwg=$result[3];
        #the version of the fw is placed at the 3rd position (starting count from 0)
                                                            }

                                                            # Query that finds the version of the cpu
fw
                                                            eval { $stm3->execute($next_fw_cpu_id); };
                                                            if ($@)
                                                            {
                                                                    print $tcpsocket "ERR 01 Server
unavailable.\r\n";
                                                                    logmsg $child_dbh->errstr;
                                                                    exit(0);
                                                            }
                                                            if (@result=$stm3->fetchrow_array)
                                                            {
                                                                    $nextfirmware_cpu=$result[3];
        #the version of the fw is placed at the 3rd position (starting count from 0)
                                                            }
                                                            $stm3->finish();
                                                            logmsg $mwg_serial.": auth OK";
                                                    }

                                            }
                                    }
################################### H001 END ###
                            }
                            else
                            {

                                    # Query that updates the OFFLINE flag
                                    $child_dbh->do("UPDATE machine_onlines SET offline=true,
connecting=false, connected=false, online=false WHERE machine_id = ?", undef, $machine_id)
                                            or die logmsg $child_dbh->errst;

                                    logmsg "DataBase machine_id: ".$machine_id."  ";

                                    logmsg "connection dropped";
                                    exit(0);
                            }
                            $count-- ;
                            if ($nextfirmware_mwg eq '')
                            {
                                    print $tcpsocket "ERR 03 S/N not authorized( $count tries
left).\r\n";
                            }
                            alarm($tcp_timeout);
                    }
################### AUTHENTICATION PROCESS END ###

                    until ( $nextfirmware_mwg ne '');
                    alarm(0);

                    $stm->finish();
                    logmsg $mwg_serial.": logged in from [".$peerhost."]";
                    print $tcpsocket "OK 02 Serial number authorized.\r\n";

                    # Query that updates the ONLINE flag
                    $child_dbh->do("UPDATE machine_onlines SET online=true, connecting=false,
connected=false, offline=false WHERE machine_id = ?", undef, $machine_id)
                            or die logmsg $child_dbh->errstr;

                    # Prepares query ID3 that finds the firmware webgate OR cpu (depending on the id
passed)
                    my $stm3 = $child_dbh->prepare("SELECT * FROM machine_firmwares WHERE version = ?")
                            or die "Can\'t prepare query 3 statement";

################### WEBGATE FIRMWARE UPDATE PROCESS BEGIN ###
                    # Prepares query ID1 that updates the fw info
```

```
                $stm = $child_dbh->prepare("UPDATE machines SET current_fw_mwg_id = ? WHERE mwg_serial
= ? AND disabled = 0")
                        or die "Can\'t prepare query 4 statement";
                $stm_ip = $child_dbh->prepare("UPDATE machines SET current_fw_mwg_id = ? WHERE
ip_address = ? AND disabled = 0")
                        or die "Can\'t prepare query 4 statement";
                alarm($tcp_timeout);
                # If the fw version has been sent AND begins with H002, goes thru
                if (defined ($firmware_mwg = <$tcpsocket>) && ($firmware_mwg =~ s/^H002\s+(.*)/$1/) )
                {
                        alarm(0);
                        $firmware_mwg =~ s/.uwg\r\n//;
                        $fw_mwg = $firmware_mwg;

                        # Executes query ID3 (see row 526) with the version sent by the webgate
                        eval { $stm3->execute($firmware_mwg); };
                        if ($@)
                        {
                                print $tcpsocket "ERR 01 Server unavailable.\r\n";
                                logmsg $child_dbh->errstr;
                                exit(0);
                        }
                        if (@result=$stm3->fetchrow_array)
                        {
                                $id = $result[0];# the ID of the fw is placed at position 0

                                # Executes query ID1 (see row 531)
                                if (($mwg_serial ne "00000000") && ($mwg_serial ne "gate.mac") &&
($mwg_serial ne "00100058") && ($mwg_serial ne "00100001") && ($mwg_serial ne "10100001"))
                                {
                                        eval { $stm->execute($id, $mwg_serial); };
                                }else{
                                        eval { $stm_ip->execute($id, $peerhost); };
                                }
                                if ($@)
                                {
                                        print $tcpsocket "ERR 01 Server unavailable.\r\n";
                                        logmsg $child_dbh->errstr;
                                        exit(0);
                                }
                                # If the flag of the fw update is false or the version on the db is
the same on the webgate sends OK 03, else tells to webgate to download the fw
                                if ((! $flag_update_fw_mwg) or ( $firmware_mwg eq
$nextfirmware_mwg.".uwg" ))
                                {
                                        print $tcpsocket "OK 03\r\n";
                                        logmsg $mwg_serial.": Fw mwg (".$firmware_mwg.") up to date";
                                }
                                else
                                {
                                        print $tcpsocket "OK 04 ".$nextfirmware_mwg.".uwg\r\n";
                                        logmsg $mwg_serial.": Fw mwg (".$firmware_mwg.") should be ".
$nextfirmware_mwg.".uwg";
                                }
                        }
                }
                else
                {
                        print $tcpsocket "ERR 04 malformed mwg firmware version\r\n";
                        logmsg $mwg_serial.": Malformed mwg firmware (version ".$firmware_mwg.")";
                }
                $stm->finish();
################### WEBGATE FIRMWARE UPDATE PROCESS END ###

################### CPU FIRMWARE UPDATE PROCESS BEGIN ###
                # Prepares query ID1 that updates the fw info
                $stm = $child_dbh->prepare("UPDATE machines SET current_fw_cpu_id = ? WHERE mwg_serial
= ? AND disabled = 0")
                        or die "Can\'t prepare query 5 statement";
                $stm_ip = $child_dbh->prepare("UPDATE machines SET current_fw_cpu_id = ? WHERE
ip_address = ? AND disabled = 0")
                        or die "Can\'t prepare query 5 statement";
                alarm($tcp_timeout);
                # If the fw version has been sent AND begins with H003 goes thru
                if (defined ($firmware_cpu = <$tcpsocket>) && ($firmware_cpu =~ s/^H003\s+(.*)/$1/) )
                {
                        alarm(0);
                        $firmware_cpu =~ s/.plc\r\n//;

                        # Executes query ID3 (see row 526) with the fw sent by webgate
                        eval { $stm3->execute($firmware_cpu); };
                        if ($@)
                        {
                                print $tcpsocket "ERR 01 Server unavailable.\r\n";
                                logmsg $child_dbh->errstr;
                                exit(0);
                        }
                        if (@result=$stm3->fetchrow_array)
                        {
                                $id = $result[0];# the ID of the fw is placed at position 0
```

```
                                    # Executes query ID1 (see row 583)
                                    if (($mwg_serial ne "00000000") && ($mwg_serial ne "gate.mac") &&
($mwg_serial ne "00100058") && ($mwg_serial ne "00100001") && ($mwg_serial ne "10100001"))
                                    {
                                            eval { $stm->execute($id, $mwg_serial); };
                                    }else{
                                            eval { $stm_ip->execute($id, $peerhost); };
                                    }
                                    if ($@)
                                    {
                                            print $tcpsocket "ERR 01 Server unavailable.\r\n";
                                            logmsg $child_dbh->errstr;
                                            exit(0);
                                    }
                                    # If the flag of the fw update is false or the version on the db is
the same on the webgate sends OK 05, else tells to webgate to download the fw
                                    if ((! $flag_update_fw_cpu) or ( $firmware_cpu eq
$nextfirmware_cpu.".plc" ))
                                    {
                                            print $tcpsocket "OK 05\r\n";
                                            logmsg $mwg_serial.": Fw cpu (".$firmware_cpu.") up to date";
                                    }
                                    else
                                    {
                                            print $tcpsocket "OK 06 ".$nextfirmware_cpu.".plc\r\n";
                                            logmsg $mwg_serial.": Fw cpu (".$firmware_cpu.") should be ".
$nextfirmware_cpu.".plc";
                                    }
                            }
                    }
                    else
                    {
                            print $tcpsocket "ERR 05 malformed cpu firmware version\r\n";
                            logmsg $mwg_serial.": Malformed cpu firmware (version ".$firmware_cpu.")";
                    }
################### CPU FIRMWARE UPDATE PROCESS END ###

                    # Closes the statements ID1 and ID3
                    $stm->finish();
                    $stm3->finish();

                    # Variable used to read the socket
                    my $linemsg;

                    alarm($tcp_timeout);

################### H006 REQUEST NUMBER OF FILES TO BE DOWNLOADED BEGIN ###
                    if ( ( substr($firmware_mwg,0,5) eq "ugate" ) && ( $flag_update_fw_mwg ||
$flag_update_fw_cpu ) )
                    {
                            # If the command sent from the webgate begins with H006 goes thru
                            if (defined ($linemsg = <$tcpsocket>) && ($linemsg =~ m/^H006/))
                            {
                                    logmsg $mwg_serial.": received H006";
                                    # Prepares the query
                                    $stm = $child_dbh->prepare("SELECT size FROM machine_firmwares WHERE
version = ?")
                                            or die "Can\'t prepare query HOO6 statement";
                                    # If the mwg firmware update flag is true executes the query with
$nextfirmware_mwg, else with $nextfirmware_cpu
                                    # WEBGATE
                                    if ( $flag_update_fw_mwg )
                                    {
                                            eval{ $stm->execute($nextfirmware_mwg); };
                                            if (@result = $stm->fetchrow_array)
                                            {
                                                    print $tcpsocket "OK 13 ".$result[0]."\r\n";
                                                    logmsg $mwg_serial.": sent OK 13 ".$result[0]." to
webgate";
                                            }
                                    }
                                    # CPU BOARD
                                    elsif ( $flag_update_fw_cpu)
                                    {
                                            eval{ $stm->execute($nextfirmware_cpu); };
                                            if (@result = $stm->fetchrow_array)
                                            {
                                                    print $tcpsocket "OK 13 ".$result[0]."\r\n";
                                                    logmsg $mwg_serial.": sent OK 13 ".$result[0]." to
webgate";
                                            }
                                    }
                            }
                            else
                            {
                                    logmsg $mwg_serial.": H006 not received!";
                            }
                    }
################### H006 REQUEST NUMBER OF FILES TO BE DOWNLOADED END ###
```

```
################### H005 MACHINE REMOTE BLOCK PROCESS BEGIN ###
                # If the command sent from the webgate begins with H005 goes thru
                if (defined ($linemsg = <$tcpsocket>) && ($linemsg =~ m/^H005/))
                {
                        logmsg $mwg_serial.": received H005";
                        print $tcpsocket "OK 11\r\n";
                        #if (QUERY FLAG IS TRUE)
                        #{
                        #       print $tcpsocket "OK 12\r\n";
                        #       logmsg $mwg_serial.": Remote machine block command sent to machine";
                        #}
                        #else
                        #{
                        #       print $tcpsocket "OK 11\r\n";
                        #       logmsg $mwg_serial.": Remote block not needed";
                        #}
                }
                else
                {
                        logmsg $mwg_serial.": H005 not received!";
                }
################### H005 MACHINE REMOTE BLOCK PROCESS END ###

                # Prepares query ID1 that inserts the snapshot
                $stm = $child_dbh->prepare("INSERT INTO machine_session_datas
(machine_session_data_type_id, machine_session_id, machine_data, machine_id, created_at) VALUES
(?, ?, ?, ?, UTC_TIMESTAMP())")
                        or die "Can\'t prepare query ID1 statement";

                # Prepares query ID FLAG_EVENTS that selects the events information
                my $stm_flag_events = $child_dbh->prepare("SELECT flag_download_events_number,
events_to_download FROM machines WHERE id = ? AND disabled = 0")
                        or die "Can\'t prepare query FLAG_EVENTS statement";

                eval{ $stm_flag_events->execute($machine_id); };

                if (@result=$stm_flag_events->fetchrow_array)
                {
                        if (($result[0] == 1) && ($result[1] != 0))
                        {
                                $events_flag_at_session_beginning = 1;
                        }elsif (($result[0] == 0) && ($result[1] == 0))
                        {
                                $events_flag_at_session_beginning = 0;
                        }
                }


                # Binds local unix socket for on-demand web connection
                my ($r_ready,$w_ready,$error,$fh,$httpclient,$linecmd,$snapshot);
                local $PLCSOCK='/tmp/PLCSOCK.';
                $PLCSOCK .= $mwg_serial;
                if ( -S $PLCSOCK )
                {
                        logmsg $mwg_serial.": Device already online. Try again later.";
                        exit(0);
                }
                unlink $PLCSOCK;

                # Creates local Unix socket
                local $httpsocket = IO::Socket::UNIX->new(
                                        Local => $PLCSOCK,
                                        Listen => SOMAXCONN,
                                        Reuse => 1);
                        die "can't setup server" unless $httpsocket;
                my $read_set = new IO::Select;
                $read_set->add($httpsocket);
                $read_set->add($tcpsocket);
                my $write_set = new IO::Select;
                $write_set->add($tcpsocket);
                logmsg $mwg_serial.": unix socket ready ($PLCSOCK)";

                # Unknown error counter, used to notify just 1 time the UNKNOWN ERROR, if occurres
                my $unknown_err_count=0;

################### CHILDLOOP BEGIN ###
CHILDLOOP:      alarm($tcp_timeout);
                while( ($r_ready,$w_ready,$error) = IO::Select->select($read_set,$write_set) )
                {
                        # Flag for the UNKNOWN ERROR
                        my $gestflag=0;

######################### FOREACH FH READ BEGIN ###
                        foreach $fh (@$r_ready)
                        {
                                if ( $fh == $httpsocket )
                                {
                                        # Check if on httpsocket there's a connection request from
httpclient.
```

```
                                                # This request is accepted and the handler of this connection
are inserted into ->select()
                                                $httpclient = $httpsocket->accept();
                                                $httpclient->autoflush(1);
                                                $read_set->add($httpclient);
                                                $write_set->add($httpclient);
                                                $gestflag=1;
                                        }
                                        elsif ( $fh == $httpclient )
                                        {
                                                # This receive data from httpclient and notify that there's
something to write into tcpsocket.
                                                if (defined ($linecmd = <$httpclient>))
                                                {
                                                        logmsg $mwg_serial.": http client sent \"".
substr($linecmd,0,64)."\"";
                                                        $write_set->add($tcpsocket);
                                                }
                                                $gestflag=1;
                                        }
                                        elsif ( $fh == $tcpsocket )
                                        {
                                                if (($firmware_mwg eq "cgate200") || ($firmware_mwg eq
"cgate201") || ($firmware_mwg eq "cgate210") || ($firmware_mwg eq "cgate211")) {
                                                        print $tcpsocket "OK 0C\r\n";
                                                        $s001_requests = $s001_requests+1;
                                                }
                                                # This check if there's something to be read from the
tcpsocket.
                                                if (defined($linemsg = <$tcpsocket>))
                                                {
                                                        if ($fw_mwg !~ m/^00.00/)
                                                        {
                                                                # Deletes the file prefix
                                                                $fw_mwg =~ s/^cgate//;
                                                                if ($fw_mwg > 145)
                                                                {
                                                                        $linemsg = <$tcpsocket>;
                                                                }
                                                        }
                                                        alarm(0);
                                                        $linemsg =~ s/\r\n//;
                                                        chomp($linemsg);
                                                        logmsg $mwg_serial.": received \"".$linemsg."\"";

                                                        if ($linemsg)
                                                        {
############################################################ S001 REGULAR SNAPSHOT BEGIN ###
                                                                if ( $linemsg =~ m/^S001/ )
                                                                {
                                                                        # If the session hasn't been
created creates it, else inserts the snapsnot
                                                                        if ($machine_session_id == '')
                                                                        {
                                                                                $machine_session_id =
&CreateSession("monitor", "snapshot", $machine_id);
                                                                        }
                                                                        else
                                                                        {
                                                                                logmsg $mwg_serial.":
Snapshot received" ;
                                                                                $snapshot=$linemsg;
                                                                                # Executes the query ID1
on row 695
                                                                                eval{ $stm-
>execute($machine_session_data_type_id, $machine_session_id, $linemsg, $machine_id); };

        &UpdateLastSnapshotId($machine_id);
                                                                        }
                                                                        if (($firmware_mwg eq "cgate200")
|| ($firmware_mwg eq "cgate201") || ($firmware_mwg eq "cgate210") || ($firmware_mwg eq "cgate211")) {
                                                                                if ($s001_requests<5) {
                                                                                        print $tcpsocket
"OK 0C\r\n";
                                                                                        $s001_requests =
$s001_requests+1;
                                                                                        logmsg
$mwg_serial.": REQUESTED S001" ;
                                                                                }
                                                                                else {
                                                                                        print $tcpsocket
"OK 0E\r\n";
                                                                                        $s001_requests =
0;
                                                                                        logmsg
$mwg_serial.": REQUESTED S003" ;
                                                                                }
                                                                        }
                                                                }
############################################################ S001 REGULAR SNAPSHOT END ###
```

```
############################################################ S002 ALARM SNAPSHOT BEGIN ###
                                                            elsif ($linemsg =~ m/^S002/)
                                                            {
                                                                    # If the session doesn't exist
creates it, else changes the type of session to alarm session
                                                                    if ($machine_session_id == '')
                                                                    {
                                                                            $machine_session_id =
&CreateSession("alarm", "snapshot", $machine_id);
                                                                    }
                                                                    else
                                                                    {
                                                                            $child_dbh->do("UPDATE
machine_sessions SET machine_session_type_id=3, timestamp_end=UTC_TIMESTAMP() WHERE id = ?", undef,
$machine_session_id)
                                                                                    or die logmsg
$child_dbh->errstr;
                                                                    }
                                                                    # If the alarm session doesn't
exist creates it and calls the email sender controller on RubyOnRails application
                                                                    if ($machine_session_alarm_id ==
'')
                                                                    {
                                                                            my $alarm_name =
substr($linemsg,95,16);
                                                                            logmsg $mwg_serial.":
Updating machine session alarms db" ;
                                                                            $machine_session_alarm_id
= &CreateSessionAlarm($machine_session_id, $alarm_name);
                                                                            logmsg $mwg_serial.":
Updating machine session datas db" ;
                                                                            # Executes the query ID1
on row 664
                                                                            eval{ $stm-
>execute($machine_session_data_type_id, $machine_session_id, $linemsg, $machine_id); };

        &UpdateLastSnapshotId($machine_id);

                                                                            # Send the OK 10 to the
mwg for notification of the received alarm only if  fw_version is >= wgate103.uwg
                                                                            # The old version
XX.YY.ZZ.uwg and the version wgate10$.uwg (where $ is 0,1 or 2) do not use the new protocol with "OK
10"
                                                                            #my $fw_mwg =
$firmware_mwg;
                                                                            #if ($fw_mwg !~ m/^00.00/)
                                                                            #{
                                                                                    # Deletes the file
extension
                                                                                    #$fw_mwg =~
s/.uwg$//;
                                                                                    # Deletes the file
prefix
                                                                                    #$fw_mwg =~
s/^wgate//;
                                                                                    #if ($fw_mwg >=
103)
                                                                                    #{
                                                                                            print
$tcpsocket "OK 10\r\n";
                                                                                            #}
                                                                                            #}

                                                                            # Uses the USER AGENT to
call che email sender on RoR application
                                                                            logmsg $mwg_serial.":
Sending alarm notification emails" ;
                                                                            my $res = $ua-
>get("http://teorema.carpigianigroup.com/machine_email/sendmail?serial_number=".
$mwg_serial."&snapshot=".$linemsg);

###########################  UNCOMMENT FOR DIALOG                   my $res = $ua-
>get("http://dialog.coldelite.it/machine_email/sendmail?serial_number=".$mwg_serial."&snapshot=".
$linemsg);
##########################
                                                                            logmsg $mwg_serial.":
Alarm notification emails sent" ;
                                                                    }
                                                            }

############################################################ S002 ALARM SNAPSHOT END ###

############################################################ S003 CURRENT PARAMETERS HASH, EVENTS AND GO
OFFLINE BEGIN ###
                                                            elsif ($linemsg =~ s/^S003(.*)/$1/ )
                                                            {
                                                                    my $stm2;
                                                                    $current_parameters_hash=$linemsg;
```

```
                                                                    logmsg $mwg_serial.": Parameters
hash received" ;
                                                                    # Executes query ID2 of row 699
                                                                    # Prepares query ID2 that selects
the parameters and hashes
                                                                    if (($mwg_serial ne "00000000") &&
($mwg_serial ne "gate.mac") && ($mwg_serial ne "00100058") && ($mwg_serial ne "00100001") &&
($mwg_serial ne "10100001"))
                                                                    {
                                                                            $stm2 = $child_dbh-
>prepare("SELECT flag_update_parameters, current_parameters_hash, next_parameters_hash, next_parameters
FROM machines WHERE mwg_serial = ? AND disabled = 0")
                                                                                    or die "Can\'t
prepare query ID2 statement";

                                                                            eval{ $stm2-
>execute($mwg_serial); };

                                                                    }else{
                                                                            $stm2 = $child_dbh-
>prepare("SELECT flag_update_parameters, current_parameters_hash, next_parameters_hash, next_parameters
FROM machines WHERE ip_address = ? AND disabled = 0")
                                                                                    or die "Can\'t
prepare query ID2 statement";

                                                                            eval{ $stm2-
>execute($peerhost); };
                                                                    }


                                                                    if( @row=$stm2->fetchrow_array() )
                                                                    {
                                                                            # If the parameters hashes
are different asks to webgate the parameters, instead goes to the else
                                                                            logmsg $mwg_serial.":
PARAMETERS => update_flag=". $row[0]." - HASHES: current=".$row[1].", next=".$row[2];
                                                                            if
( ($current_parameters_hash ne "") and ($current_parameters_hash ne $row[1] ) )
                                                                            {
                                                                                    print $tcpsocket
"OK 08\r\n";
                                                                                    logmsg
$mwg_serial.": Request current parameters" ;
                                                                            }
                                                                            else
                                                                            {
                                                                                    # If the flag is
true sends the next parameters to webgate, else checks for events and go offline flag
                                                                                    if( $row[0] )
                                                                                    {
                                                                                            if
( $current_parameters_hash eq $row[2] )
                                                                                            {

        logmsg $mwg_serial.": Params MD5 up2date" ;

        if (($mwg_serial ne "00000000") && ($mwg_serial ne "gate.mac") && ($mwg_serial ne "00100058")
&& ($mwg_serial ne "00100001") && ($mwg_serial ne "10100001"))
                                                                                                    {

        $child_dbh->do( q{UPDATE machines SET current_parameters_hash = next_parameters_hash ,
current_parameters = next_parameters , current_parameters_timestamp = UTC_TIMESTAMP() WHERE mwg_serial
= ? AND disabled = 0 } , undef, $mwg_serial);
                                                                                                    }
else{

        $child_dbh->do( q{UPDATE machines SET current_parameters_hash = next_parameters_hash ,
current_parameters = next_parameters , current_parameters_timestamp = UTC_TIMESTAMP() WHERE ip_address
= ? AND disabled = 0 } , undef, $peerhost);
                                                                                                    }
                                                                                            }
                                                                                            else
                                                                                            {

        logmsg $mwg_serial.": Sending next parameters" . $row[3]. "-" ;

        print $tcpsocket "OK 09".$row[3]."\r\n";

                                                                                            }
                                                                                    }
                                                                                    else
                                                                                    {
                                                                                            #
Executes the flag ID FLAG EVENTS on row 703
                                                                                            logmsg
$mwg_serial.": Params MD5 up2date" ;

        eval{ $stm_flag_events->execute($machine_id); };

                                                                                            if
(@result=$stm_flag_events->fetchrow_array)
```

```
{
#
Sends EV 01 to webgate if the flag is true and the number of events to download is zero
        if (($result[0] == 1) && ($result[1] == 0))
{
        # Send EV 01 to webgate to start the download of events.
        logmsg $mwg_serial.": EVENTS flag_download_events_number is true, send EV01 to WG" ;
        print $tcpsocket "EV 01\r\n";
}
#
Sends EV 02 to webgate if the flag is true and the number of events to download is not zero
        elsif (($result[0] == 1) && ($result[1] != 0))
{
        if ($events_flag_at_session_beginning == 1)
        {
                my $last_connection = $child_dbh->prepare("SELECT max(id) FROM machine_sessions WHERE
id != ? AND machine_id = ?")
                        or die "Can\'t prepare query FLAG statement";
                eval { $last_connection->execute($machine_session_id, $machine_id); };
                if (@result_conn=$last_connection->fetchrow_array)
                {
                        my $max_id = $result_conn[0];
                        $last_connection = $child_dbh->prepare("SELECT timestamp_begin FROM
machine_sessions WHERE id = ?")
                                or die "Can\'t prepare query FLAG statement";
                        eval {$last_connection->execute($max_id); };
                        if (@result_conn=$last_connection->fetchrow_array)
                        {
                                $last_snapshot_timestamp = $result_conn[0];
                        }

                }
                # Check the timestamp of the "Start Download Events" pressure:
                # if it is passed more than 15 minutes set the flags to false,
                # else do nothing.

                my $end_events = DateTime->now( time_zone => 'floating' );
                my $fmt = '%Y-%m-%d %H:%M:%S';
                my $parser = DateTime::Format::Strptime->new(
                        pattern => $fmt,
                        time_zone => 'GMT',
                        );


                my $dt1 = $parser->parse_datetime($last_snapshot_timestamp) or die;
                #my $dt2 = $parser->parse_datetime($end_events) or die;
                #my $diff = $dt2 - $dt1;

                my $diff = $end_events - $dt1;

                if ($diff->minutes >= 15)
                {
```

```
                        logmsg $diff->minutes." from the last session with the download events flag =
1 => set the flags = 0 and do not download events";

                        # Set flags for download events to false.

                        $child_dbh->do("UPDATE machines SET flag_download_events_number=0,
events_to_download=0, available_events=0 WHERE id = ?", undef, $machine_id)

                                or die logmsg $child_dbh->errstr;

                }else{

                        logmsg $mwg_serial.": EVENTS number to download: ".$result[1] ;

                        print $tcpsocket "EV 02 $result[1]\r\n";

                }

        }else{

                logmsg $mwg_serial.": EVENTS number to download: ".$result[1] ;

                print $tcpsocket "EV 02 $result[1]\r\n";

        }
                                                                                                        }
                                                                                                }

                                                                                                # If the
go offline flag is true tells to webgate to go offline
                                                                                                logmsg
$mwg_serial.": MWG go offline?" ;
                                                                                                my
$stm_flag = $child_dbh->prepare("SELECT go_offline FROM machines WHERE id = ?")

        or die "Can\'t prepare query FLAG statement";
                                                                                                eval
{ $stm_flag->execute($machine_id); };
                                                                                                if
(@result=$stm_flag->fetchrow_array)
                                                                                                {

        $go_offline=$result[0];
                                                                                                }
                                                                                                if
($go_offline==0)
                                                                                                {

        logmsg $mwg_serial.": FLAG is ".$go_offline." => MWG stays online" ;

        print $tcpsocket "OK 07\r\n";
                                                                                                }
                                                                                                else
                                                                                                {

        logmsg $mwg_serial.": FLAG is ".$go_offline." => MWG goes offline" ;
                                                                                                        #
Check if we have received events during the current session.
                                                                                                        #
If so set to false the download events flag,
                                                                                                        #
if not do nothing.

        if ($event_session_started == 1)
                                                                                                        {

        $child_dbh->do("UPDATE machines SET flag_download_events_number=0, available_events=0,
events_to_download=0 WHERE id = ?", undef, $machine_id)

                or die logmsg $child_dbh->errstr;

        $event_session_started = 0;
                                                                                                        }

        print $tcpsocket "OK 0A\r\n";   # Was OK 11

        $child_dbh->do("UPDATE machines SET go_offline=false WHERE id = ?", undef, $machine_id)

        or die logmsg $child_dbh->errstr
                                                                                                }
                                                                                        }
                                                                                }
                                                                        }
                                                                        else
                                                                        {
                                                                                logmsg $mwg_serial.": SQL
problem ".$child_dbh->errstr; ;
```

```
                                                                            print $tcpsocket "OK
07\r\n";
                                                                            logmsg $mwg_serial.":
Params infos up2date" ;
                                                                    }
                                                            }
########################################################### S003 CURRENT PARAMETERS HASH, EVENTS AND GO
OFFLINE END ###

########################################################### S004 CURRENT PARAMETERS BEGIN ###
                                                            elsif ($linemsg =~ s/^S004(.*)/$1/ )
                                                            {
                                                                    $current_parameters = $linemsg;
                                                                    #if ($fw_mwg !~ m/^00.00/)
                                                                    #{
                                                                            # Deletes the file prefix
                                                                            #$fw_mwg =~ s/^cgate//;
                                                                            if (($firmware_mwg eq
"cgate160") || ($firmware_mwg eq "cgate170") || ($firmware_mwg eq "cgate161") || ($firmware_mwg eq
"cgate171") || ($firmware_mwg eq "cgate200") || ($firmware_mwg eq "cgate201") || ($firmware_mwg eq
"cgate210") || ($firmware_mwg eq "cgate211") || ($firmware_mwg eq "cgate220") || ($firmware_mwg eq
"cgate221") || ($firmware_mwg eq "cgate230") || ($firmware_mwg eq "cgate260") || ($firmware_mwg eq
"cgate360") || ($firmware_mwg eq "cgate180") || ($firmware_mwg eq "ugate146") || ($firmware_mwg eq
"cgate460") || ($firmware_mwg eq "cgate560") || ($firmware_mwg eq "ugate200"))
                                                                            {
                                                                                    ## From fw version
1.60 it is necessary to look for the \0 at the end of the parameters table in
                                                             ## order to calculate the correct md5

                                                                                    logmsg
$mwg_serial.": Nuovo metodo di calcolo MD5 $firmware_mwg";
                                                                                    my $index = rindex
$current_parameters, "\0";

        $current_parameters = (substr $current_parameters, 0, $index);
                                                                            }
                                                                            $current_parameters_hash =
Digest::MD5::md5_hex ($current_parameters);
                                                                            #}

                                                                    logmsg $mwg_serial.": MD5
calcolato: $current_parameters_hash";
                                                                    if (($mwg_serial ne "00000000") &&
($mwg_serial ne "gate.mac") && ($mwg_serial ne "00100058") && ($mwg_serial ne "00100001") &&
($mwg_serial ne "10100001"))
                                                                    {
                                                                            $child_dbh->do( q{UPDATE
machines SET flag_update_parameters=false, current_parameters_hash = ? , current_parameters = ? ,
current_parameters_timestamp = UTC_TIMESTAMP() , next_parameters_hash = ? , next_parameters = ? ,
next_parameters_timestamp = UTC_TIMESTAMP() WHERE mwg_serial = ? AND disabled = 0 }, undef,
$current_parameters_hash, $current_parameters, $current_parameters_hash, $current_parameters,
$mwg_serial);
                                                                    }else{
                                                                            $child_dbh->do( q{UPDATE
machines SET flag_update_parameters=false, current_parameters_hash = ? , current_parameters = ? ,
current_parameters_timestamp = UTC_TIMESTAMP() , next_parameters_hash = ? , next_parameters = ? ,
next_parameters_timestamp = UTC_TIMESTAMP() WHERE ip_address = ? AND disabled = 0 }, undef,
$current_parameters_hash, $current_parameters, $current_parameters_hash, $current_parameters,
$peerhost);
                                                                    }
                                                                    # Executes query ID1 of row 695
                                                                    eval{ $stm->execute(2,
$machine_session_id, $current_parameters, $machine_id); };
                                                            }
########################################################### S004 CURRENT PARAMETERS END ###

########################################################### S005 AUTOMATIC SESSION BEGIN ###
                                                            elsif ($linemsg =~ /^S005/)
                                                            {
                                                                    logmsg $mwg_serial.": Received S005
=> AUTOMATIC SESSION RECOGNIZED";
                                                                    $auto = 1;
                                                                    # Set flags for download events to
true.
                                                                    $child_dbh->do("UPDATE machines SET
flag_download_events_number=1, events_to_download=0, available_events=0 WHERE id = ?", undef,
$machine_id)
                                                                            or die logmsg $child_dbh-
>errstr;
                                                                    logmsg $mwg_serial.": Send EV01 to
Webgate";
                                                                    print $tcpsocket "EV 01\r\n";
                                                            }
########################################################### S005 AUTOMATIC SESSION END ###

########################################################### S101 AVAILABLE EVENTS NUMBER BEGIN ###
                                                            elsif ($linemsg =~ s/^S101(.*)/$1/ )
                                                            {
                                                                    logmsg $mwg_serial.": Received S101
- Number of available events: ".$1 ;
```

```
							$last_available_events = $1;
							$child_dbh->do("UPDATE machines SET 
available_events = ?, last_event_read=UTC_TIMESTAMP(), flag_download_events_number=false WHERE id = ?", 
undef, $last_available_events, $machine_id)
								or die logmsg $child_dbh-
>errstr;
							# If the automatic session flag is 
false goes thru, instead sends the EV02 command
							if($auto==1)
							{
								logmsg $mwg_serial.": 
automatic sending of EVENTS number to download: ".$1 ;
								print $tcpsocket "EV 02 
$1\r\n";
								logmsg "EV 02 SENT\n";
								$child_dbh->do("UPDATE 
machines SET flag_download_events_number=true, events_to_download=? WHERE id=?", undef, $1, 
$machine_id)
									or die logmsg 
$child_dbh->errstr;
### NOT NEEDED							$child_dbh->do("UPDATE 
machines SET flag_download_events_number=false WHERE id=?", undef, $machine_id)
### NOT NEEDED								or die logmsg 
$child_dbh->errstr;
							}
						}
########################################################## S101 AVAILABLE EVENTS NUMBER END ###

########################################################## S102 STORE EVENT BEGIN ###
						elsif ($linemsg =~ s/^S102(.*)/$1/ )
						{
							if ($event_session_id=='')
							{
								$event_session_id = 
&CreateEventSession($machine_id);
							}
							$event_session_started = 1;

							logmsg $mwg_serial.": Received S102 
- Event number ".substr($1,0,10)." of ".substr($1,10,10) ;
							$events = $1;
							$child_dbh->do("INSERT INTO events 
(event_session_id, event_data, created_at, machine_id) VALUES (?,?,UTC_TIMESTAMP(),?)", undef, 
$event_session_id, $events, $machine_id)
								or die logmsg $child_dbh-
>errstr;
						}
########################################################## S102 STORE EVENT END ###

########################################################## S103 END EVENTS DOWNLOAD BEGIN ###
						elsif ($linemsg =~ /^S103/ )
						{
							logmsg $mwg_serial.": Received 
S103" ;
							$child_dbh->do("UPDATE machines SET 
available_events=0, events_to_download=0, flag_download_events_number=0 WHERE id = ?", undef, 
$machine_id)
								or die logmsg $child_dbh-
>errstr;
							$child_dbh->do("UPDATE 
event_sessions SET updated_at=UTC_TIMESTAMP() WHERE id=?", undef, $event_session_id)
								or die logmsg $child_dbh-
>errstr;
							if($auto==1)
							{
								logmsg $mwg_serial.": 
events received after automatic connection. Webgate now goes offline.";
								$child_dbh->do("UPDATE 
machines SET go_offline=true WHERE id=?", undef, $machine_id)
									or die logmsg 
$child_dbh->errstr;
							}
							next;
						}
########################################################## S103 END EVENTS DOWNLOAD END ###

					}
					alarm($tcp_timeout);
				}
				else
				{	# If enters here it received undefined data (such as 
the FYN packet at the end of TCP connection)

					# Closes the session only if opened
					if ($machine_session_id != '')
					{
						$child_dbh->do("UPDATE machine_sessions SET 
timestamp_end=UTC_TIMESTAMP() WHERE id = ?", undef, $machine_session_id)
							or die logmsg $child_dbh->errstr;
					}
```

```
						# Closes the events download session only if opened
						if ($event_session_id != '')
						{
							$child_dbh->do("UPDATE machines SET 
available_events=0, events_to_download=0 WHERE id = ?", undef, $machine_id)
								or die logmsg $child_dbh->errstr;
							$child_dbh->do("UPDATE event_sessions SET 
updated_at=UTC_TIMESTAMP() WHERE id=?", undef, $event_session_id)
								or die logmsg $child_dbh->errstr;
						}

						# Sets flag_update_parameters = false
						if (($mwg_serial ne "00000000") && ($mwg_serial ne 
"gate.mac") && ($mwg_serial ne "00100058") && ($mwg_serial ne "00100001") && ($mwg_serial ne 
"10100001"))
						{
							$child_dbh->do("UPDATE machines SET 
flag_update_parameters=false WHERE mwg_serial = ? AND disabled = 0", undef, $mwg_serial)
									or die logmsg $child_dbh-
>errstr;
						}else{
							$child_dbh->do("UPDATE machines SET 
flag_update_parameters=false WHERE ip_address = ? AND disabled = 0", undef, $peerhost)
									or die logmsg $child_dbh-
>errstr;
						}

						logmsg $mwg_serial.": TCP connection dropped";

						# Updates the OFFLINE flag
						$child_dbh->do("UPDATE machine_onlines SET 
last_session_id=?, offline=true, connecting=false, connected=false, online=false WHERE machine_id = ?", 
undef, $machine_session_id, $machine_id)
							or die logmsg $child_dbh->errstr;

						$read_set->remove($tcpsocket);
						if (defined ($httpclient)) {
							$read_set->remove($httpclient);
							$write_set->remove($httpclient);
							$httpclient->close;
						}
						$read_set->remove($httpsocket);
						$httpsocket->close;
						unlink $PLCSOCK;
						# Exit child
						logmsg "Child exiting...";
						exit(0);
						last CHILDLOOP;		# this is a break of all 
code-block identified by CHILDLOOP (il ciclo while). It includes the sig_alrm handler.
					}
					$gestflag=1;
				}
			}
########################### FOREACH FH READ END ###

########################### FOREACH FH WRITE BEGIN ###
			foreach $fh (@$w_ready){
				if ( $fh == $httpclient ) {
					# Writes the snapshot received from tcpsocket and send to 
httpclient
					if (defined($snapshot)) {
						print $httpclient $snapshot;
					} else {
						print $httpclient "SNAPSHOT NOT AVAILABLE";
					}
					$write_set->remove($httpclient);
					$read_set->remove($httpclient);
					$httpclient->close;
					$gestflag=1;
				} elsif ( $fh == $tcpsocket ) {
					# Writes on the tcpsocket the data previously read from 
httpclient.
					if ( defined($linecmd) && ($linecmd != "" )) {
						logmsg "PLC TCP socket want to read unix socket";
						print $tcpsocket $linecmd;
					}
					$write_set->remove($tcpsocket);
					$gestflag=1;
				}
			}
########################### FOREACH FH WRITE END ###

########################### FOREACH FH ERROR BEGIN ###
			foreach $fh(@$error){
				logmsg "SELECT ERR";
				$gestflag=1;
			}
########################### FOREACH FH ERROR END ###
```

```
                        # Writes on the logfile the UNKNOWN EVENT ERROR, only one time per session
                        if ( $unknown_err_count ==  0 ) {
                                logmsg "??? unknown event\n" unless ($gestflag == 1);
                                $unknown_err_count++;
                        }
                }
################### CHILDLOOP END ###

                # Used to safely execute the daemon in case of exit from the main loop
                exit(0);
        }
        else
        {
                # PARENT process closes unused handler and go back to accept()
                logmsg "INET socket opened, forking...";
                close($tcpsocket);
        }
########### FORK END ###
}
### PARENT FOR CYCLE END ###
```

List of Publications

The publications related to the main contributions of this thesis are stated as follows.

Proceedings

R. Lazzarini, C. Stefanelli, M. Tortonesi - Large-scale e-Maintenance: A new frontier for management? - Proceedings of 2013 IFIP/IEEE International Symposium on Integrated Network Management (IM 2013) - 732-735, 2013.

R. Lazzarini, C. Stefanelli, M. Tortonesi, G. Virgilli - e-Maintenance of Carpigiani Ice Cream Making Machines - Proceedings of Congresso Nazionale AICA 2011, Turin, Italy - 15-17 November 2011.

R. Lazzarini, C. Stefanelli, M. Tortonesi, G. Virgilli - Teorema: an e-Maintenance Platform for Ice Cream Machines - Proceedings of 16th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA 2011), Toulouse, France - 5-9 September 2011.

M. Grossi, M. Lanzoni, A. Pompei, R. Lazzarini, D. Matteuzzi, B. Riccò - A portable biosensor system for bacterial concentration measurements in cow's raw milk - Proceedings of the 4th Interantional Workshop on Advences in Sensors and Interfaces (IWASI) Savelletri di Fasano, Italy - 132-137, June 2011.

Book Chapters

R. Lazzarini, C. Stefanelli, M. Tortonesi - Leveraging ICT to Enable e-Maintenance for Automated Machines - Adaptive, Dynamic, and Resilient Systems, CRC Press, Taylor & Francis Group, Boca Raton, Florida, USA - 215-236, 2014.

Articles Int. Journals

A. Sardone, R. Lazzarini, B. Zauli, J. Ronco, R. Pelosi - L'eco-innovazione nell'industria manifatturiera italiana: due casi di successo - Energia, Ambiente e Innovazione, ENEA (DOI:10.12910/EAI2013-20) – Vol. 2013, No. 5, September-October 2013.

M. Grossi, R. Lazzarini, M. Lanzoni, A. Pompei, D. Matteuzzi, B. Riccò - A Portable Sensor with Disposable Electrodes for Water Bacterial Quality Assessment - IEEE Sensor Journal (ISSN1530-437X) - Vol. 13, No. 5, 1775-1782, May 2013.

R. Lazzarini, C. Stefanelli, M. Tortonesi, G. Virgilli - e-Maintenance for Household and Similar Appliances - International Journal of Productivity and Quality Management, Inderscience (ISSN1746-6474) - 12 (2), 141-160, 2013.

M. Grossi, M. Lanzoni, R. Lazzarini, B. Riccò - Automatic ice-cream characterization by impedance measurements for optimal machine setting - Measurement, Elsevier (ISSN0263-2241) - Vol. 45, No. 7, 1747-1754, August 2012.

A. Pompei, M. Grossi, M. Lanzoni, G. Perretti, R. Lazzarini, B. Riccò, D. Matteuzzi - Feasibility of Lactobacilli Concentration Detection in Beer by Automated Impedance Technique - Technical Quarterly 49, Master Brewers Association of the Americas (MBAA) - Vol. 49, No. 1, 11-18, 2012.

M. Grossi, M. Lanzoni, R. Lazzarini, B. Riccò - Linear Non Iterative Sinusoidal Fitting Algorithm for Microbial Impedance Biosensor - Sensors & Transducers Journal, International Frequency Sensor Association (IFSA) - Vol. 137, No. 2, 235-244, February 2012.

L. Saraceno, G. Boccardi, G.P. Celata, R. Lazzarini, R. Trinchieri - Development of two heat transfer correlations for a scraped surface heat exchanger in an ice-cream machine - Applied Thermal Engineering, Elsevier (ISSN 1359-4311) - Vol. 31, No. 17, 4106-4112, December 2011.

M. Grossi, R. Lazzarini, M. Lanzoni e B. Riccò – A novel technique to control ice cream freezing by electrical characteristics analysis - Journal of Food Engineering, Elsevier (ISSN0260-8774) - Vol. 106, No. 4, 347-354, October 2011.

Patents

R. Lazzarini, A. Cocchi, "Macchina e metodo per la produzione di prodotti liquidi o semiliquidi provvista di inverter comandato mediante un bus di campo", Italian Patent BO2014A000670, 28 November 2014.

A. Cocchi, M. Tortonesi, C. Stefanelli, R. Lazzarini, “Method and apparatus for making and dispensing liquid or semi-liquid food products”, Japanese Patent JP2014-193159, 9 October 2014.

A. Cocchi, M. Tortonesi, C. Stefanelli, R. Lazzarini, “Method and apparatus for making and dispensing liquid or semi-liquid food products”, United States Patent US2014/0295044, 2 October 2014.

A. Cocchi, M. Tortonesi, C. Stefanelli, R. Lazzarini, “Method and apparatus for making and dispensing liquid or semi-liquid food products”, Chinese Patent Application CN104068198, 1 October 2014.

A. Cocchi, M. Tortonesi, C. Stefanelli, R. Lazzarini, “Method and apparatus for making and dispensing liquid or semi-liquid food products”, European Patent EP2783574, 1 October 2014.

A. Cocchi, M. Tortonesi, C. Stefanelli, R. Lazzarini, "Metodo ed apparecchiatura per la produzione ed erogazione di prodotti alimentari liquidi o semiliquidi", Italian Patent BO2013A000134, 28 March 2013.

Acknowledgement

This work and all that it represents for me would not have been possible without the help and support of Cesare Stefanelli and Mauro Tortonesi, of the University of Ferrara, and Andrea Cocchi and Roberto Morisi of Carpigiani Group.