TAMPEREEN TEKNILLINEN YLIOPISTO
TAMPERE UNIVERSITY OF TECHNOLOGY

GOVINDARAJAN NAVEEN KUMAR

DEVELOPING FUNCTION BLOCKS FOR COLLECTING DATA AND INTEGRATING LEGACY SYSTEMS IN MANUFACTURING AND LOGISTICS

Master of Science thesis

Examiner: Prof. Jose L. Martinez Lastra
Examiner and topic approved by the Council meeting of the Faculty of Engineering Sciences on 28th February 2018

ABSTRACT

NAVEEN KUMAR GOVINDARAJAN
Tampere University of technology
Master of Science Thesis, 64 pages
April 2018
Master's Degree Programme in Automation Engineering
Major: Factory Automation and Industrial Informatics
Examiner: Prof. Jose L. Martinez Lastra

Keywords: Manufacturing, Data Collection, IEC61499, Enterprise Integration, Distributed Automation, Service-Oriented-Architecture, Legacy System.

Manufacturing enterprises have been increasingly technology-driven during recent decades. Industry 4.0 promotes smart manufacturing and intelligent systems which can seamlessly communicate with each other and enable decentralized decision-making by monitoring the factory-floor process. This calls for the Information Communication Technologies (ICT) infrastructure to be effectively incorporated with the industries. Industry 4.0 presents the concept of "smart factory" in which Cyber-Physical-Systems (CPS) fuses the physical systems with the Internet of Things (IoT), enabling higher levels of interoperability and Information transparency.

However, manufacturing enterprises in the recent past have characterized their efficiency by how prominently and adequately they adopt and utilize their IT solutions and how feasible those solutions are to integrate with their legacy systems. Enterprise Integration, in particular, has become more challenging owing to the highly dynamic manufacturing environment. System integration has become an indispensable field to be addressed , especially when the industry adopts connected enterprise paradigm. Connected enterprise systems enable industries to leverage their technologies to collect, analyze and refine their data to help them make better business decisions.

In a recent trend, IT systems in manufacturing are majorly driven towards the cloud and collaborative solutions as a result of the exponential growth of internet technologies and their ability to adapt to rapid changes in the market. Collaborative frameworks are widely preferred by the enterprises as they enable better communication, increases productivity and improve business execution. They are critical for a business to function with agility in this fast pacing and changing world. One such platform is provided by the Cloud Collaborative Manufacturing Networks (C2NET) project that optimizes the supply network of manufacturing and logistics assets.

This thesis research proposes an approach to integrate heterogeneous legacy systems by showcasing an implementation which favors robust data collection. This implementation is made possible by adopting Production Logistics and Sustainability Cockpit (PLANTCockpit) Open Source solution, which functions as a viable interface for real-time data collection and data-logistics thus enhancing the process optimization of the manufacturing enterprise. PLANTCockpit OS is a modular solution which enables to build and deploy flexible loosely coupled entities known as Function Blocks (FBs) that facilitate seamless legacy system integration and robust information exchange between the systems. This thesis also fulfills the C2NET project requirement to define the possibility of effective integration of PLANTCockpit OS in the C2NET reference architecture.

PREFACE

The underlying thesis research would have been an impossible task to accomplish without earning the constant support, guidance, and motivation in various diverse ways. I feel that it is appropriate to acknowledge and recognize the special people who helped me to reap the benefits of my strenuous labor.

At this moment, I express my sincere gratefulness to prof. José Luis Martínez Lastra for the opportunity to study in TUT and work as a Research Assistant at FAST-Lab to enhance my skills in factory automation domain, which ultimately led to the inception of this thesis. I am thankful for his impeccable scientific guidance and his passion for teaching and sharing knowledge. I thank Anne Korhonen and Matti Aarnio for making me feel comfortable and welcome during my time in FAST-Lab.

I sincerely extend my gratitude to my supervisor Dr. Borja Ramis Ferrer who has been with me since day one of this thesis and guiding me on the right track towards perfection. His incredible motivational capabilities along with interdisciplinary cooperation were significantly valuable to me. I also want to thank Sergii, Xu, Angelica and Dr. Andrei Lobov who have been instrumental in mentoring and motivating me and above all, showing persistent confidence in my abilities.

I consider myself extremely fortunate to have earned many friends during my stay in TUT who have been extraordinary and supportive. Special thanks to Balaji, Karthik, Vivek, Shriram, Sathish, Nirmal, Arsalan, and Ahsan. And the enlightening coffee sessions with Amir, Luis, and Mohammed during those crucial times of my thesis will not be forgotten.

I devotedly thank my beloved parents and my little sister who has always been there for me, who has always been proud of me no matter what. They all have been of great support to me both morally and emotionally. This journey of mine would have been challenging if not for their awesomeness and encouragement.

Finally, I genuinely dedicate my greatest possible thankfulness to Shalini, who has been the motivational factor for me to complete my Master's degree and my thesis. Her perpetual love and immaculate affection was the paramount element that kept me going every single time. Her unconditional inspiration and unbelievable care were pivotal as it gave me the strength to achieve this feat.

Thank you for everything.

Naveen Kumar Govindarajan

14th April 2018

Tampere, Finland

CONTENTS

LIST OF FIGURES

LIST OF SYMBOLS AND ABBREVIATIONS

BC	Business configuration
C2NET	Cloud Collaborative Manufacturing Networks
CPS	Cyber-Physical Systems
DCAP	Distributed Control Application Platform
DCC	Data Collection Client
DPWS	Device Profile for Web Services
EA	Enterprise Architecture
EI	Enterprise Integration
ERP	Enterprise Resource Planning
ESB	Enterprise Service Bus
FB	Function Block
FBEC	Function Block Engine Configurator
FBN	Function Block Network
FIS	Factory Information Systems
GDP	Gross domestic product
ICT	Information and Communication Technology
IEC	International Electro-Technical Commission
IEC	International Electro-Technical Commission
IoT	Internet of Things
JMS	Java Message Service
JSON	JavaScript Object Notation
MES	Manufacturing Execution System
MOM	Manufacturing Operations Management
OI	Operational Iintelligence systems
OSGi	Open Service Gateway Initiative
PID	Proportional-Integral-Derivative
PLANTCockpit	Production Logistics and Sustainability Cockpit
PLC	Programmable Logic Controllers
R&D	Research and Development
REST	Representational State Transfer
SCADA	Supervisory Control and data acquisition
SOA	Service-oriented architecture
XML	Extensible Markup Language

1. INTRODUCTION

This section focuses on providing the motivation, background and the purpose of this thesis. Also, this section introduces the problem definition and the problem statement addressed in this thesis. This section justifies the work done which answers the question as to why this topic/work is essential. Finally, Section 1 concludes by providing a brief introduction to the forthcoming sections in this thesis.

1.1 Background and Motivation for the work

Manufacturing engineering came into existence with the modernization and transformations of socio-economic standards during the 18-19th century. This is broadly characterized as the first industrial revolution [1] which first originated in Britain that later widely spread to rest of the world over the course of time. Manufacturing industries as a whole, contribute a significant proportion to a country's economy. These industries employ a wide range of technologies and process management methodologies [2]. The manufacturing industries in the European union incorporates a broad scope of methods and production approaches, ranging from small and medium scale enterprises (SME) to large enterprises including the companies that manufacture raw materials that also comprised of commodities or components.

Manufacturing is a crucial entity that determines a country's Gross domestic product (GDP). In 2014, around 9% of all the industries in the 28 states in the EU are characterized under manufacturing sector [3]. With a total of 2.1 million enterprises, manufacturing sector contributes to 22.1% of employment workforce and is forecasted to boost for the foreseeable future. Among the diverse subsectors of manufacturing, the most significant subsector in the EU regarding value added and employment in the manufacturing enterprise of machinery and equipment [4]. This is possible by EU's flagship initiative Horizon 2020 [5], which tackles Research & Innovation and drives the economic growth in the global market and is considered as EU's blueprint for making them vital players in manufacturing smart and sustainable systems.

The industrial revolution has seen enormous advancements/modernization over the past two centuries since its inception. Based on the manufacturing statistics exhibited in [6], The European union's rising trend in the industrial production and manufacturing sector has completely recovered from its steep downfall from the economic crisis of 2008 [6]. We are at present in the 4th Industrial revolution [7], which is popularized and characterized as Industry 4.0 and is the generation of smart manufacturing [8], which primarily involves making the machines interact with each other over the network. This dynamic approach can be made possible by centralizing the operational data and incorporating the process data into the physical systems to make them as a single entity. This prominent approach is closely associated with Information and Communication Technology (ICT), along with Internet of Things (IoT) systems, Cyber-Physical Systems (CPS) and Enterprise Integration (EI) [9]. Industry 4.0 aspires to meet the agile and dynamic requirements of the manufacturing industry and to reach a greater level of operational efficiency and productivity. The prospective of manufacturing industries relies on integrating the physical systems with the information technology infrastructure, favoring data analytics. The contribution of data analytics is seen throughout the product lifecycle which enhances the optimization of the process. Continuous data collection methods from a variety of sources and then integrating them to the systems in the manufacturing industry in real-time has begun to gain popularity in the recent past. One such type of data acquired by making a connection between the legacy systems through continuous data collection methods can also be proved useful to optimize production planning [10].

The Master of Sciences Thesis supports the research activities of Tampere University of Technology, precisely at the Factory Automation and Systems Technology Laboratory (FAST-Lab). Some of the research activities of the FAST-Lab are strongly connected with the industry through a series of projects funded by the European Commission. With the increase in industrial revolution and modernization, every industry needs to collect and process data from the manufacturing systems, to enhance production efficiency and managing product quality. This thesis work can be considered as an extension of [11].

The primary purpose of collecting data is to feed the received data to other systems, usually for optimizing the system as a whole. Data collection is done in numerous stages of business process, the data collected should be consistent and standardized which ensures that the data is accessible all through the industry. The consolidated and comprehensive framework which defines the complete data collection methodology and enables optimized data acquisition can be termed as Data Collection Framework.

For a business to continually improve in the market, the business levels of production and process of that enterprise must be integrated. Integration leads to reduce costs and increase flexibility, finally ending up with on-time delivery and improvement in quality. One of the primary limitations in the industry is absence of seamless integration of information flow between Vertical and Horizontal levels of the industry. Vertical integration denotes integration of information flow right from low-level applications towards high-level applications and vice versa along the vertical direction in the manufacturing industry hierarchy [12].

1.2 Problem Definition

Nowadays, manufacturing industries demands modern advancements to minimize the complexity in deploying the enterprise systems. However, the majority of the small and medium scale (SME) industries due to their limited resources, do not have access to procure cutting-edge manufacturing systems and tools and hence cannot afford to establish enterprise system integration.

Manufacturing industries contribute enormously to the growth of an economy and possess a vast potential to improve the domestic Research and Development (R&D) [13]. This being said, manufacturing industries face an increasing number of complications in responding and adapting to rapid changes in the market, keeping up with the requirements of the client and consumers and rising labor costs [14].

This expanding complexity in the manufacturing processes continually demands for development of more robust, efficient, and flexible systems that facilitate seamless enterprise integration. In the manufacturing industry, an extensive amount of data is generated from various data sources ranging from paper, excel sheets to sensors, IoT devices, and machines, etc. This data-driven manufacturing approach leads to empower next generation of manufacturing solutions and strengthen production decision-making process [15].

A significant process in manufacturing industries relies on collecting the shop floor data from contemporary Supervisory Control systems and various data sources. These data are then processed in the Manufacturing Execution Systems (MES) and the resulting business process is handled in Enterprise Resource Planning (ERP). However, more often, these crucial data are manually fed to these systems. To eradicate the possible shortcomings (untimeliness, inaccuracy, and bias, etc.) of data collection, continuous and automated data collection systems are vital to managing information flow throughout the enterprise efficiently.

1.2.1 Problem statement

Owing to the rise in technologies, seamless enterprise integration, data collection and data logistics are vital. By employing flexible enterprise integration systems, the manufacturing industries can not only reduce costs but at the same time improve and enhance their efficiency and productivity. The objective above can be attained by answering the below questions:

- How to effectively utilize PLANTCockpit OS solution to build FB Adapters which act as an interface to enterprise systems?
- How to deploy and initiate an interconnected network of entities (FBs) which facilitate data collection?
- How does the solution permit efficient Data collection with heterogeneous legacy systems?
- How does the framework as a whole, facilitate seamless Enterprise Integration?

1.3 Work description

1.3.1 Research Objectives

The objective of this research is listed as follows:

- To present a FunctionBlock (FB) approach which empowers robust data collection and data logistics by employing PLANTCockpit OS solution. FBs are loosely coupled entities with execution algorithms encapsulated in them.
- To evaluate the essential characteristics of the enterprise systems and the defining the requirements for the development of the needed interfaces (FB Adapters) for PLANTCockpit platform.
- To fully define, develop, and deploy the needed Function Blocks and Adapters from the conceptual and technical point of view and tested in real scenarios.
- To deploy a network of FBs using PLANTCockpit OS platform within a distributed environment. An integral part of this research is the implementation carried out by demonstrating the solution in an industrial use case scenario.
- To validate the individual components deployed in the use case scenario. The PLANTCockpit OS platform will integrate the information collected from a network of interconnected industrial systems, and this task will require the validation of individual component's lifecycle and their results.

1.3.2 Research Methodology

The research was proceeded after a structured review of different open source and commercial data collection frameworks for "easy" integration of data sources in the industrial domain as well as a study of the most common information systems in factories (e.g., ERPs, MES, SCADA). An analysis of the results from the data collection solution implemented is reported, as well as a study with the open source integration tool for data collection (PLANTCockpit-OS) was made to examine as to how the solution can be used to prove this research and how efficient would the integration of it into the cloud platform would perform.

Also, a personal evaluation of the different industrial systems from a set of C2NET industrial pilots will was made. This investigation paved way for defining the primary interfaces that must be developed for the data collection solution using the PLANTCockpit-OS. This solution can be considered as a proof of concept which would permit PLANTCockpit OS later to be integrated with the C2NET platform, which has as the primary target to support the optimization in data collection of manufacturing and logistics assets.

1.4 Thesis Outline

The thesis document is classified into 6 Sections. The Section 2 presents some theoretical knowledge about manufacturing systems available, techniques and methodologies deployed in factory automation. This section also mentions some literature review of data collection and data collection framework related techniques implemented in the manufacturing industries.

Section 3 illustrates some insights on the approach used to support the implementation of this approach. It provides a broader perspective of how the data collection methodology utilizing the Function block technique is approached. Section 3 presents the overview of the tools, techniques, and technologies that were utilized to implement the approach.

Section 4 presents the proposed solution for Legacy systems integration and data collection using Function blocks and provides a detailed description of how they work. This section also tends to offer the in-depth view of the implementation of Function blocks along with their business logic. The vital scope of this section is the implementation of the use case as discussed in the previous chapter and providing a detailed analysis of the implementation and describing the effectiveness if this use case.

Section 5 exhibits the use case implementation results. It describes the output obtained from the individual function blocks and also the output from the consolidated use case as a whole. This section highlights the flexibility of Function blocks and to prove its simple and effective.

Section 6 addresses the problem statements that was solved which were made possible with the proposed solution and highlighted the benefits of using Function block adapters. This section also provides suggestions and guidelines for improvements to be made to the future to implement them in a broader perspective.

1.5 Research Limitations

Despite the fact that the objective of this research has attained, there are few assumptions and limitations:

- Although this thesis concentrates on implementing a solution which facilitates flexible Enterprise Integration, it does not take into consideration the security of the consolidated framework. Consideration of security aspects was not the focus of this thesis research.
- Even though the Function blocks designed and implemented are robust, for the thesis, only four different type of function blocks are designed and implemented to prove the use case.

- Industries in reality, employ a wide range of data sources, but only REST, Excel and SQL data sources are focused in this research. FB Adapters act as an interface to these data sources to collect and transport the data.
- Also, the data formats in these function blocks are restricted to XML and JSON as it is sufficient to serve the purpose of proving the use case/solution.

2. THEORETICAL BACKGROUND

This section introduces some theoretical knowledge on the Manufacturing systems. This section also tends to explain the standards that contribute to Enterprise system integration. This section also sheds some light on systems that employ data collection, along with terminologies that facilitate robust distributed industrial automation. The integral part of this section would give an overview of the framework which is extensively used for system integration.

2.1 Manufacturing systems

Modern technological advancements have benefited the manufactures by providing productivity, improvised asset utilization and enhanced decision-making. This evolution is driven by the competence of integrating the information technology systems with business process systems, enabling enterprises to leverage their shop floor data.

In a critical financial circumstance, manufacturing industries need to adjust to economic situations and settle on choices that enhance revenue for the organization. According to [16], the author states that in countries that have a thriving economy, it is crucial to increase the productivity to compete with the demand. Likewise, in countries that are low in the economy, it is vital to decrease costs and meet the demand. To obtain this information, industries implement various real-time data acquisition solutions such as Operational intelligence (OI) systems, Factory Information Systems (FIS), Manufacturing Execution Systems and Manufacturing Operations Management systems (MES & MOMs).

Operational intelligence solutions provide real-time information from across IT systems and technology infrastructure which facilitates efficient decision making [17]. More complex manufacturing operations (multisite or multi-plant operations for instance) require operational intelligence solutions that enable them to manage the complexities of the factories.

Manufacturing Execution Systems (MES) [18] is a comprehensive framework that provides complete process transparency between business process and the factory-floor. MES is a state-of-the-art tool that takes process monitoring to a whole new level and optimizes the production output. MES architecture can be used for autonomous data collection and information flow across all the layers of the enterprise [19]. MES, due to its high-quality product traceability, can be seamlessly incorporated to work with enterprise ERP systems as well. MES can be applied in situations where it becomes necessary for events to be triggered by specific predetermined actions. MES solutions are used for well-operation of the industry reliably. MES aims to provide services such as real-time data acquisition to increase the efficiency of the production process [20].

MES functionalities can be well understood using the MESA model. Manufacturing Execution Solutions Association(MESA) [21], is a well-renowned association that has adapted MES framework extensively and promotes innovation and best practices to deliver valuable knowledge of enterprise systems. MESA characterizes the industry standard methodologies that enables industries to optimize their business process execution with enterprise-level requirements [22]. A pragmatic approach of MESA also known as collaborative MES or (c-MES) in short is illustrated in the figure below. This methodology defines twelve function groups to assist efficient production management.

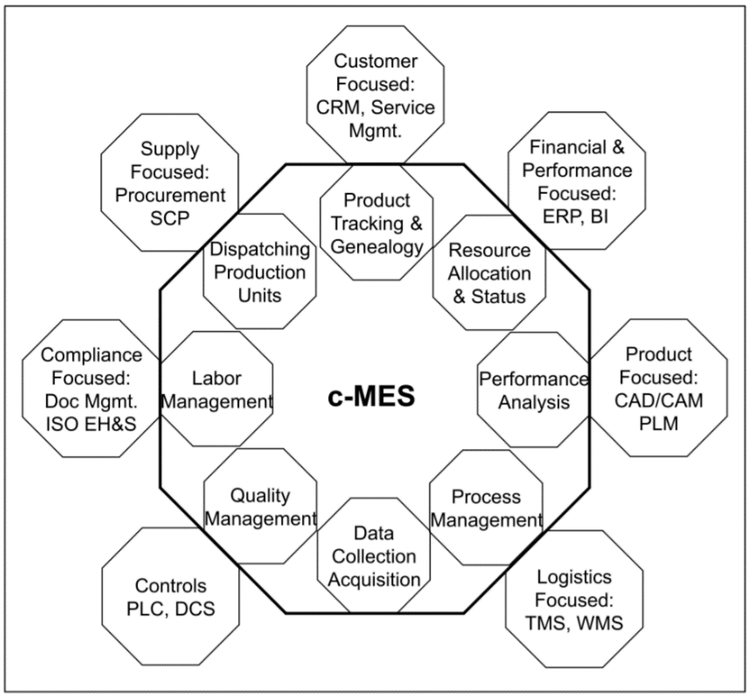

Customer Focused: CRM, Service Mgmt.
Supply Focused: Procurement SCP
Financial & Performance Focused: ERP, BI
Product Tracking & Genealogy
Dispatching Production Units
Resource Allocation & Status
Compliance Focused: Doc Mgmt. ISO EH&S
Labor Management
c-MES
Performance Analysis
Product Focused: CAD/CAM PLM
Quality Management
Process Management
Data Collection Acquisition
Controls PLC, DCS
Logistics Focused: TMS, WMS

Figure 1. The function structure of the MESA model [18]

Manufacturing Operations Management (MOM) [23] is a comprehensive solution that provides complete transparency in manufacturing operations which optimizes the efficiency of the performance of the manufacturing processes. MES & MOM are frequently interchangeable terms. MOM provides a holistic framework that integrates diverse functionalities of the manufacturing industry by incorporating MES functionalities at its core [24]. Making the manufacturing process digitize, enhances the flexibility of the organization and also facilitate the industry to rapidly adapt and respond to frequent market changes and radical innovations [25]. The below illustration highlights the position and functionalities of Enterprise systems, and it can be visually observed that MES functions as a compelling interface layer actively interchanging information between the process control systems and planning systems.

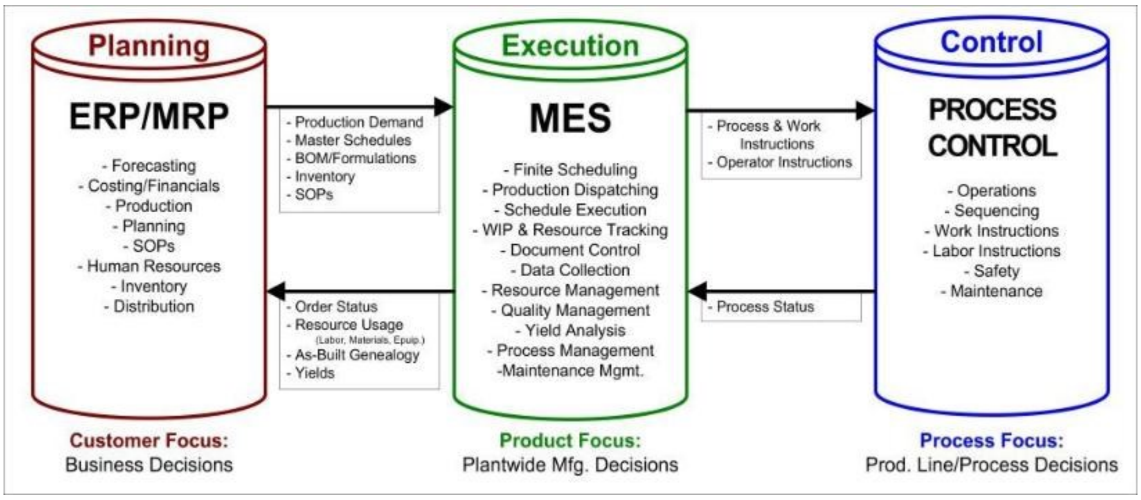


Figure 2. Manufacturing Systems information flow [26]

Every industry should have a stable means to collect and manage data. By Advancing the Data Acquisition methods and modernizing the information infrastructure, the organization has the potential to revolutionize its Application domain [27]. One of the essential components of such infrastructures is the Data Management systems that facilitate seamless integration of data that are spread across levels in the organization ranging from a diverse range of data sources to large enterprise Databases where the data are stored for further use.

Enterprise integration is a specialized technical area of Enterprise Architecture, which concentrates on the interconnection of systems, electronic data interchange, exchanging supply chain of product data and distributed computing environments. It is an approach in Enterprise engineering which aims to provide crucial and right information during the execution empowering seamless and robust communication between individuals, machines, and PCs and facilitate active co-operation and co-ordination between the systems. Data Collection Framework offers enterprises to tap into heterogeneous data sources which in turn promote Vertical/Horizontal integration at the different level of enterprise system integration.

Vertical Integration leads to higher flexibility of the data [28]. While considering vertical integration, Data and communication flow in the process-oriented enterprise should comply with the levels of the business pyramid as described in [28], it implies the flow of data from higher level systems towards the lower level systems and vice-versa [12]. To enhance Data Integration in industrial systems, the concept of Horizontal Integration method is adapted. Horizontal integration method provides users to execute complex data transformations originating from several data sources [12].

2.2 Data collection

2.2.1 The significance of Data Collection

The definition of data can take up many forms depending on the context being used [29]. As per the most commonly used terminology, data is defined as "*factual statistics used as a basis for reasoning, discussion, or calculation"* [30]. Information can also take many discipline-specific definitions. All processes produce one generic definition being *"information is the value within the outcome of any process"* [31].

Data collection in general can be characterized as the process of acquiring and compiling information from. The information gathered and compiled might come from heterogeneous data sources. Agile data collection is considered as an integral part of almost every manufacturing industry that is inclining towards lean production [32]. The formalized good practice of data collection process ought to be built up, adjusted to suit the circumstances, and audited occasionally.

Data collection methods are essential for discovering and handling existing data and additionally to generate new data [33]. Other procedures where data collection comes into play include maintaining the flow of data, enhancing estimations, and replacing existing data sources when those right now utilized are no more accessible. Data collection procedures should be chosen in such a way that it can iteratively enhance the quality of the inventory by the data quality objectives. While collecting data, Researchers must keep in mind specific requirements such as which data to be collected, how to manage the data, who will receive the data and finally When to collect the data.

2.2.2 Data Collection Framework

Manufacturing industries nowadays are majorly driven by modernization and globalization. Owing to the unstable market and shorter product life cycles, every manufacturing industry needs to collect and process data to enhance production efficiency and manage product quality. This can be made possible with the utilization of industrial Data Acquisition and data processing systems [34]. As the name suggests, Data collection refers to a systematic approach to collecting data from various data sources to interpret, analyze and make them more structured. Data collection is carried out to evaluate the results obtained and making them available to the data consumers [10]. The figure below presents a transparent data collection system model integrated with the MES system which is capable of optimizing the business process.

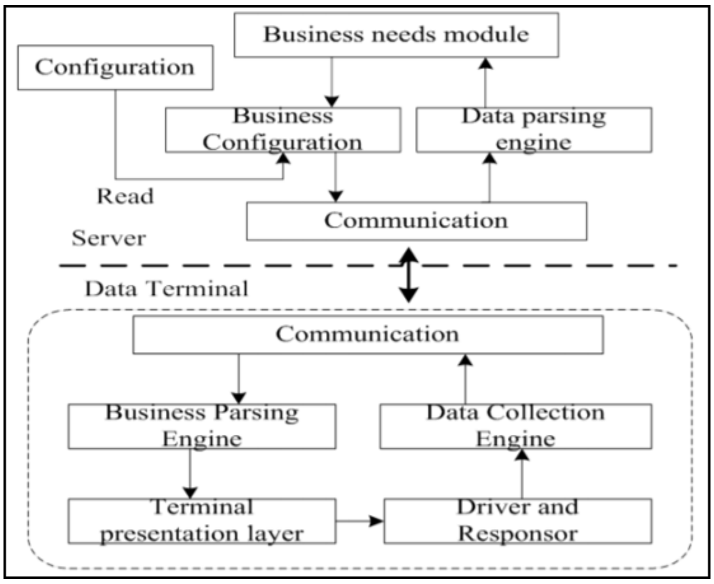


Figure 3. Data collection system model in manufacturing [35]

Efficiency and product quality plays a vital role in modernization and globalization of industry. Organizations must work around and enhance these factors to stay competitive in today's volatile markets. To achieve this goal, optimization of production process becomes inevitable. It is crucial to optimize the production processes, in this manner expanding efficiency while maintaining or improving product quality. Be that as it may, the accomplishment of these objectives requires precise and accurate data. Continuous Real-Time data originated from the industry must be accessible at all times. This is the best way to construct a solid foundation for all further optimization processes.

Data Collection Framework consists of various individual elements that encompass to form one single component. Data Collection Framework has technical aspects to be taken into consideration such as communication protocol between the data, data parsing methods, data filtering/aggregation procedure, streaming data analysis, data storage, data transformations, data security and cloud computing, etc. The Organization needs data to let it know how business is performing. Manufacturing industries need to adapt to economic situations and should make decisions that are profitable to the company.

2.2.3 Real-time continuous Data Collection in Manufacturing

Data Collection methodologies can have different approaches to different equipment, and the architecture of that particular methodology can have an exclusively unique design [36]. The next step after collecting data is to store the data collected in an accessible form. The data collected are stored in files, and they can be accessed whenever required. Another critical aspect which needs to be considered is, parsing of the contents of the data collected. Data parsers are to be designed and configured in par with the requirements of each equipment to obtain the desired data which are intended to be collected.

Every industry can have their own unique sets of data and data collection methods. The data collection methods also work on individual protocols by which, different sets of data obtained from various devices can be treated independently. To improve the real-time efficiency of the manufacturing industry, interoperability of data and data analytics solutions should be considered as the primary focus [37], which includes manufacturing data management that comprises of data acquisition obtained throughout the supply chain of manufacturing stages [38].

Real-time Data collection may lead to accumulation of the enormous amount of data. For an industry to sustain, these vast volumes of data need to be maintained and managed efficiently. Data integration and data interoperability significantly influence organizations performance. On account of utilization of a diverse form of data, data integration and data interoperability will pose a complicated threat to the manufacturing industries in the field of data collection [39].

Article [40] presents us a solution to solve the issue of collecting heterogeneous data and integrating them into industries by extracting metadata. [27] presents an Adaptive Data Collection method for distributed environments that adapts meticulously to frequently changing sensor values as a result of application requirements also optimizing the sensor's energy consumption. On the other hand, [41] shows a scalable, adaptive data collection technique that acquires data from all sensor nodes within the Wireless sensor networks.

2.3 Enterprise System Integration

Enterprise integration enables industries to tackle the mission-critical concern of automating and streamlining the business process and making the systems more Robust, flexible and interoperable [42]. In the book [43], the author addresses the Enterprise integration with a futuristic view as an integral tool to enhance business processes. The principal focus is on establishing the connection between the systems and the business processes and precisely how efficient it is to build an interconnected and integrated application framework which implements the desired business processes.

This modularization of Enterprise Integration can be achieved with a significant level of maturity by adapting asynchronous messaging and communication within the application systems and by following the principles that are associated with Service-oriented architecture [44]. The author handles this concept, by developing a layered bottom-up approach starting from the application-oriented-approach in the bottom layer, preceded by the service-oriented integration in the middle tier and finally completed by the process-oriented integration which sits at the topmost level which can be depicted in the figure below.

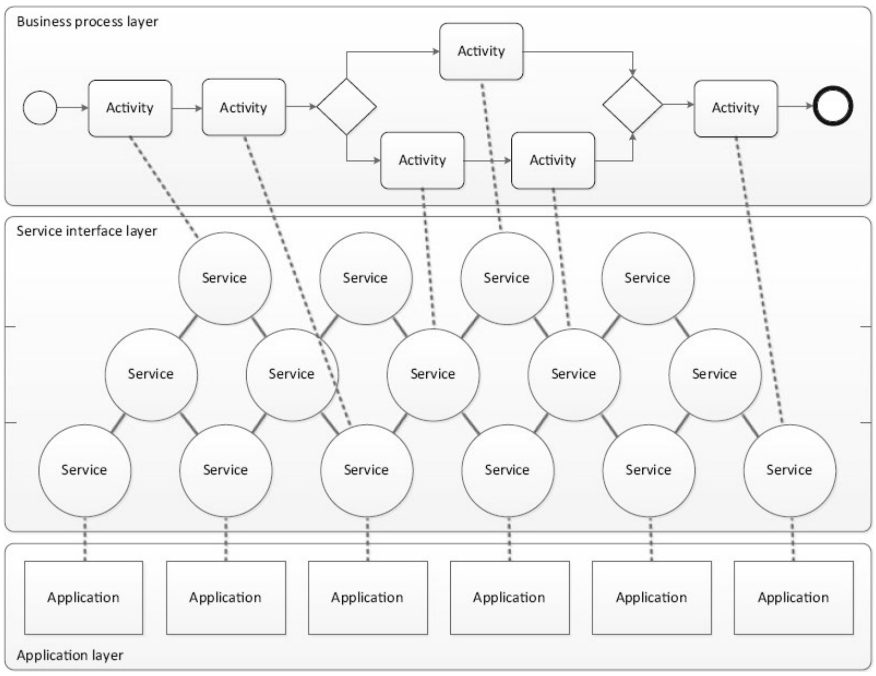


Figure 4. Enterprise Integration layers [44]

However, the implementation of Enterprise Integration can be very cumbersome due to the technologies encompassing the environments are constantly changing [45]. To yield the benefits of the distributed and modular systems, the industry must implement the techniques which address the problems faced by the existing framework. Some of the most common challenges faced by the industries and which should be dealt with are Interoperability, Data Integration, Robustness, Stability and Scalability [46] - [47].

2.3.1 Automation Pyramid

The conventional Automation pyramid model [48] offers a series of well-structured layers to that reduces the complexity of information flow across all the layers of Manufacturing industry. The integration of technologies is well exhibited across the five layers of Automation pyramid as they interrelate between each level using industrial communication methods [49]. The proper definition of all the automation pyramid layers is vital and essential while implementing the vertical integration of business levels in production or process-oriented enterprise.

Level -1: The base level also called the “field level” or the “device level” corresponds to physical devices and data sources at the factory floor. These devices include sensors, actuators, and hardware.

Level -2: The second level can also be termed as the "control level." consists of the logical devices and machine interfaces namely Desktop Computers, Programmable Logic Controllers (PLC), Proportional-Integral-Derivative (PID), etc.

Level -3: The third "supervisory level" comprises of Supervisory Control systems, characterized as SCADA, which permits the operators to monitor and control the systems.

Level-4: The Manufacturing Execution System is one of the higher level systems which takes care of monitoring the manufacturing and business process execution .

Level-5: The Enterprise resource planning system resides at the top of the pyramid which is considered as the management layer.

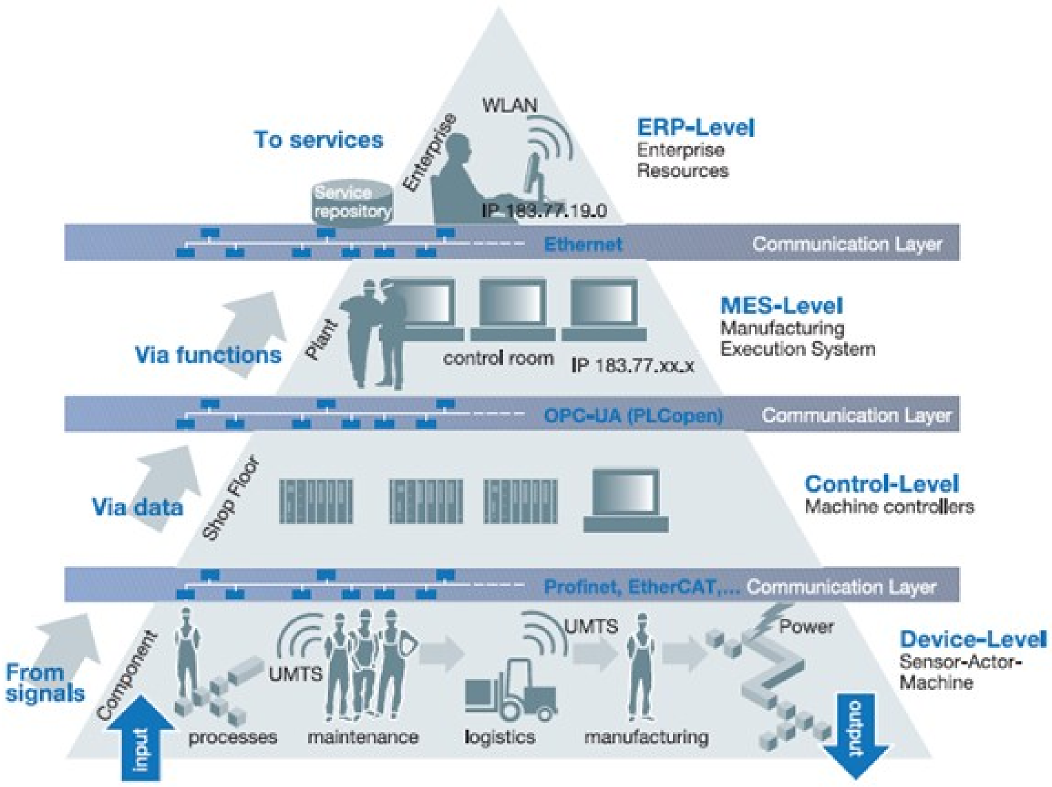


Figure 5. The layered architecture of Automation Pyramid [50]

2.3.2 ISA-95 Standard

The international standard ISA-95 [51] was conceived to facilitate the integration of enterprise and control systems. This international standard incorporates building an autonomous self-sufficient interface which seamlessly integrates enterprise and control systems. The ISA-95 standard is characterized by defining the components and operations involved in manufacturing and business levels by bridging the gap between ERP and Manufacturing Execution Systems (MES) [52].

ISA-95 standard is categorized into five influential parts which help us to address the complexities encountered exchange of information across all the layers of hierarchy within in the industry.

Part 1 (ANSI/ISA 95.01): **Models and Terminology** [53] comprises standard terminologies and fuses the object models that defines a boundary between the enterprise systems and control system and guides to determine the information that needs to be exchanged between applications.

Part 2 (ANSI/ISA 95.02): **Object Model Attributes** [54] accounts for the attributes assigned to individual object defined in the previous part. This part is responsible for exchanging information between different systems.

Part 3 (ANSI/ISA 95.03): **Activity Models** [55] focuses on the functions and information flows on the MES layer. This layer depicts the model reference to an extensive variety of activities ranging from production and quality to maintenance and inventory.

Part 4 (ANSI/ISA 95.04): **Object Models & Attributes** [56] represents the object models that defines the exchange of information between various MES activities that were discussed in Part 3.

Part 5 (ANSI/ISA 95.05): **Business to Manufacturing (B2M) transactions** [57] showcases the exchange of information and the transactions between the enterprise business and the production automation systems and vice versa.

The ISA-95 framework addresses several terminologies which reduce the complexity of enterprise integration of business logistics systems and manufacturing systems [58] by providing a framework that enables seamless information flow between enterprise management systems and manufacturing operation systems. Hence, the ISA-95 standard can be considered as an answer to the interoperability requirements between distinct levels of an enterprise [59]. Below figure presents the functional levels that perform a discrete set of activities of an industrial system as defined by the ISA-95 standard.

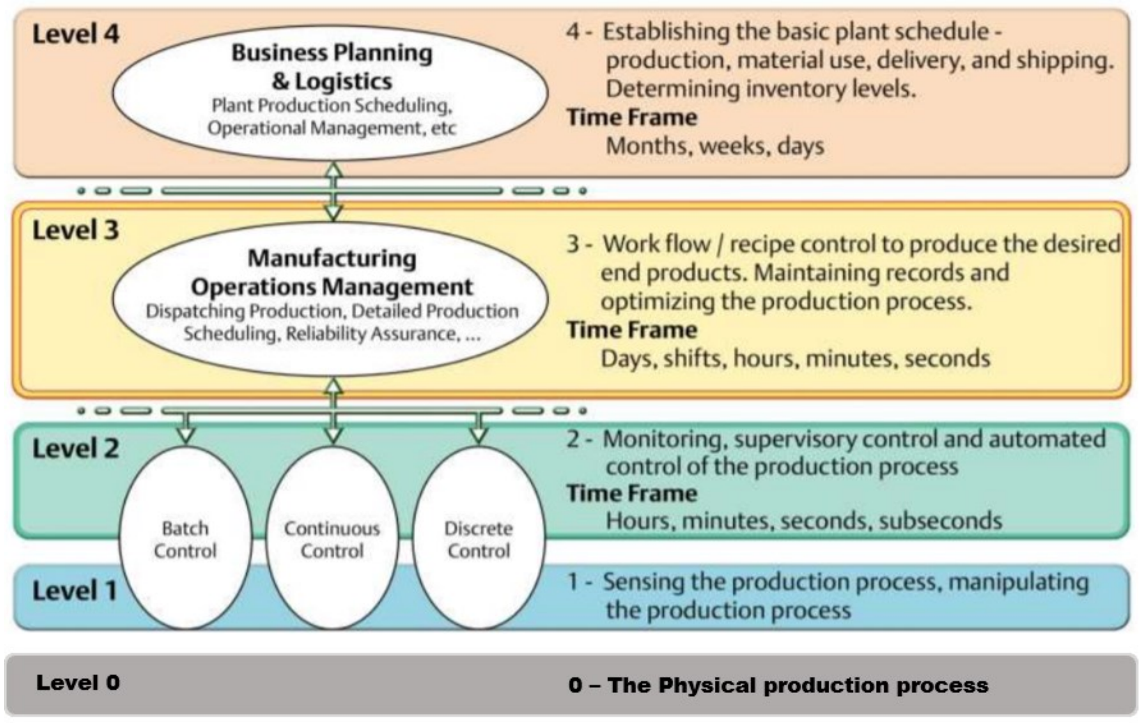

Level 4
Business Planning & Logistics
Plant Production Scheduling, Operational Management, etc
4 - Establishing the basic plant schedule - production, material use, delivery, and shipping. Determining inventory levels.
Time Frame
Months, weeks, days
Level 3
Manufacturing Operations Management
Dispatching Production, Detailed Production Scheduling, Reliability Assurance, ...
3 - Work flow / recipe control to produce the desired end products. Maintaining records and optimizing the production process.
Time Frame
Days, shifts, hours, minutes, seconds
Level 2
Batch Control
Continuous Control
Discrete Control
2 - Monitoring, supervisory control and automated control of the production process
Time Frame
Hours, minutes, seconds, subseconds
Level 1
1 - Sensing the production process, manipulating the production process
Level 0
0 – The Physical production process

Figure 6. ISA-95 Control Hierarchy Level s[60]

Manufacturing industries in the recent times have massively invested in the ERP and SCADA systems to enhance the process layer automation. MES serves as an interface which fills the gap that exists between the business automation layer and process automation layer. The unique functionalities and interactions defined in the ISA-95 standard are adapted and characterized in the Purdue Reference Model [61] in the figure below. The dotted line shown in the picture represents the Level 3 and Level 4 of the ISA-95 interface which illustrates the boundary between ERP and the enterprise control systems. When taken a closer look at the structure, it can be deduced that almost all the systems within the dotted radius area are typical MES systems [48].

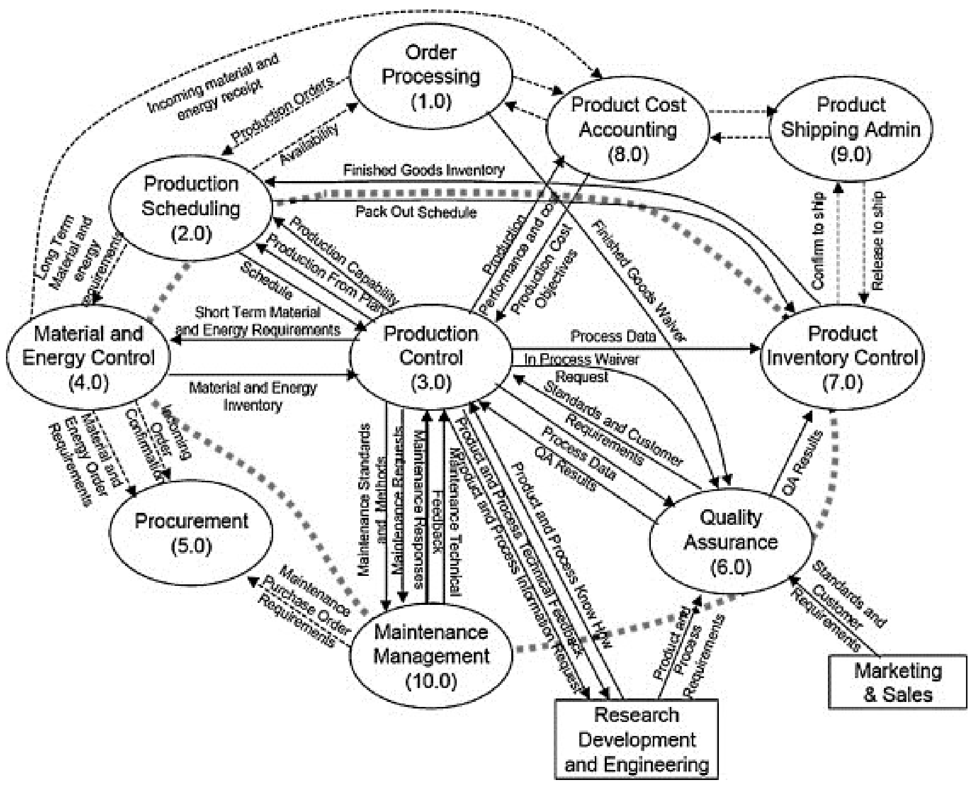

Order Processing (1.0)
Production Scheduling (2.0)
Production Control (3.0)
Material and Energy Control (4.0)
Procurement (5.0)
Quality Assurance (6.0)
Product Inventory Control (7.0)
Product Cost Accounting (8.0)
Product Shipping Admin (9.0)
Maintenance Management (10.0)
Research Development and Engineering
Marketing & Sales
Incoming material and energy receipt
Production Orders
Availability
Finished Goods Inventory
Pack Out Schedule
Long Term Material and energy requirements
Production Capability
Production From Plan
Schedule
Short Term Material and Energy Requirements
Material and Energy Inventory
Process Data
In Process Waiver Request
Standards and Customer Requirements
Process Data
QA Results
Production Performance and costs
Production Cost Objectives
Finished Goods Waiver
Confirm to ship
Release to ship
Material and Energy Order Requirements
Order Confirmation
Incoming
Maintenance Purchase Order Requirements
Maintenance Standards and Methods
Maintenance Requests
Maintenance Responses
Feedback
Maintenance Technical
Product and Process Information Request
Product and Process Technical Feedback
Product and Process Know How
Product and Process Requirements
QA Results
Standards and Customer Requirements

Figure 7. Functional enterprise-control model [62]

2.3.3 Industrial Legacy Systems integration

The present is continually evolving environment demands better industrial grade systems. But not all the enterprises have the resources to upgrade to new systems now and then. So, the wise decision would be to make the best use of the existing legacy systems and be able to manage wide volumes of data and process them. Legacy systems [63] are classified as the outdated enterprise systems being employed in the industry despite available upgraded systems.

Legacy systems are referred to the computer systems inclusive of both hardware and software that also involves programming languages and application programs or any other terminologies that either obsolete or out of date, but are still being used in the industry as they prove to be business critical systems [64]. Age does not necessarily classify legacy systems. The term legacy may also indicate lack of vendor support and the system's incompetence to meet the organization's requirements [65].

Organizations always tend to appeal to the ever-increasing demands to modernize legacy systems. Modernizing the legacy systems permits them to be interoperable with other internal or external systems. A comprehensive system evolution approach incorporating an enterprise framework to upgrade legacy systems has been addressed in [66]. On the other hand, Jha et al., 2014 in [67] present a systematic approach whose core goal is to integrate Big Data solutions into enterprise legacy systems which focuses to enable rapid and real-time data interpretation by incorporating Big Data to diverse data sources.

Manufacturing industries are always on the lookout for advanced, cutting-edge solutions that enables the enterprise to increase the range and efficiency of enterprise system integration in the industry. This empowers the industry to create dynamic systems that are mature enough to adapt themselves with the existing legacy systems and also with an extensive variety of new frameworks. Therefore, it is also vital to implement stable and efficient legacy systems so that seamless integration with future technology is ensured. Qifeng in [68] provides a semantic framework model which facilitates intelligent integration of heterogeneous information systems.

2.4 Distributed Industrial Automation

Distributed systems take up various nomenclature. According to Steen and Tanenbaum in [69], a distributed system can be loosely characterized as a collection of interconnected systems to form one consolidated system that facilitates flexible automation environment by efficient exchange of messages and information flow. In principle, Distributed systems are collective computing systems that comprise of nodes that are capable of working autonomously by seamless collaboration. A node can be classified as both a hardware system or a software process that functions independently. Distributed systems are often standardized as an overlay network [70] which enables collection of nodes to communicate with each other to form a single coherent system. The figure below depicts a simplified distributed system coordinated as middleware which extends over multiple networked machines and provides a similar interface for each system and application.

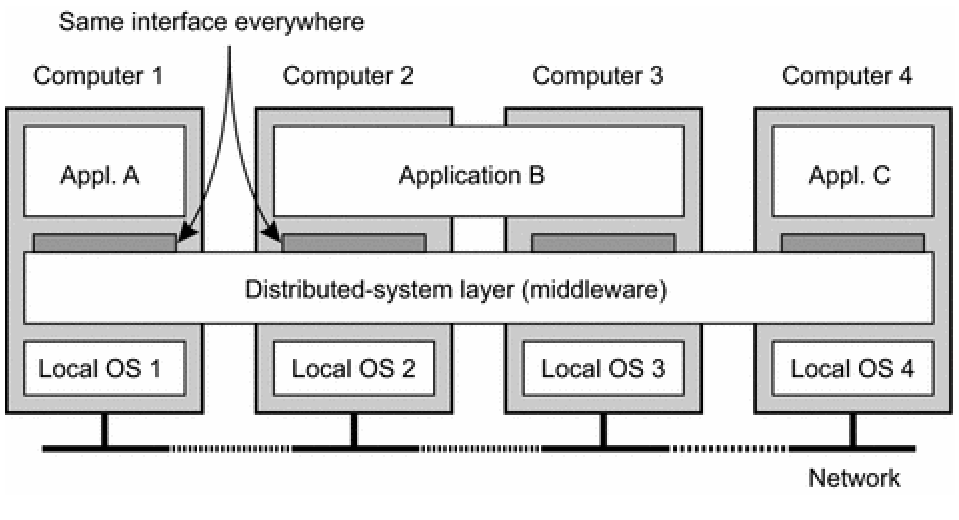


Figure 8. A distributed system organized as middleware [69]

In modern-day manufacturing, the distributed systems are mission-critical systems to interact, collaborate and ensure flexibility, adaptability, and robustness of the system. These complex systems consist of various interconnected subsystems or nodes that represent individual functions such as automation control systems, MES and ERP systems and other plant-related systems. That authors in [71], presents a multi-agent based industrial control systems that incorporate pattern recognition feature which simplifies event processing and management. Helo et al. [72] points out the needs and challenges of distributed manufacturing and proposes a next-generation cloud-based MES solution architecture to overcome the shortfall.

Interoperability is one of the crucial shortcomings of distributed systems. Based on Makki's observation [73], most legacy systems lack interoperability with the legacy systems. The author also provides a more in-depth analysis of integration of monolithic legacy systems into distributed systems. The figure below interprets the underlying representation of the Distributes system architecture that incorporates enterprise application integration and ensures that same communication framework can be used to interact with legacy and distributed systems.

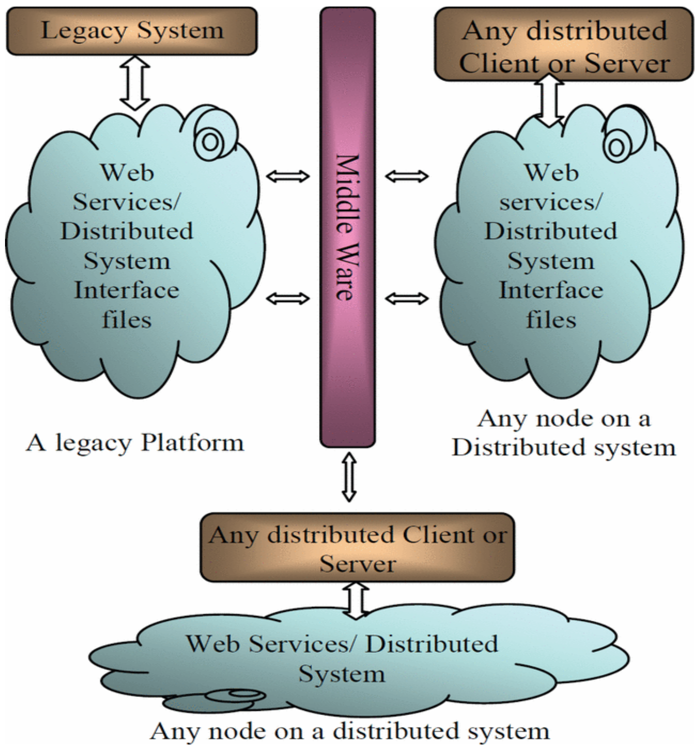

Legacy System
Web Services/ Distributed System Interface files
A legacy Platform
Middle Ware
Any distributed Client or Server
Web services/ Distributed System Interface files
Any node on a Distributed system
Any distributed Client or Server
Web Services/ Distributed System
Any node on a distributed system

Figure 9. Basic Distributed System Architecture [73]

2.4.1 Service-Oriented-Architecture

The Service-oriented-architecture (SOA) paradigm can be termed as a transformation of distributed computing that empowers industries to design asynchronous and synchronous systems and applications [74]. SOA can be conceptualized as a framework for constructing integrated and loosely coupled applications. SOA is capable of building interoperable and distributed enterprise applications by adopting function block techniques [75]. Service-oriented-architecture is modular, operating across heterogeneous operating systems, cross-platform applications, system software and function across an entire enterprise infrastructure [76].

Contemporary enterprise systems are frequently evolving from monolithic systems to agile distributed systems with service-oriented flexible capabilities. Service-oriented applications in some scenarios can be associated with event-driven systems [77] and these both paradigms are modularized and aims at maximizing the reusability and flexibility of the systems thus increasing the adaptability and efficiency of the system [78]. Service-oriented systems enable the industries to adapt their legacy systems to meet the rapidly changing market demands [79]. A simple representation and workflow of SOA can be observed in the figure below.

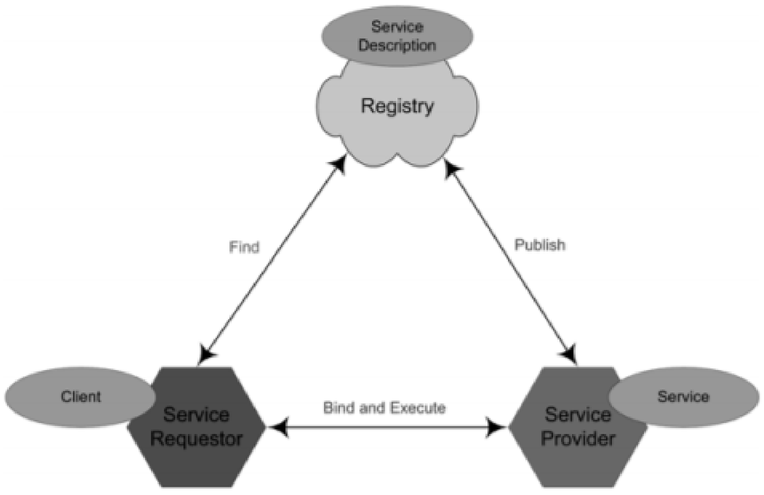


Figure 10. Service-Oriented Architecture [80]

2.4.2 IEC-61499 Standard

The International Electro-technical Commission (IEC) has now conceived a new standard IEC61499 [81] which now empowers the function blocks to encapsulate distributed industrial automation solutions. IEC 61499 standard, characterizes universal guidelines for the employment of function blocks in distributed industrial processes and control systems. The adaptation of the IEC 61499 standard helps to solve the complexity of enterprise integration by providing high reusability and reconfigurability along with the portability of applications, which enhances the scalability, flexibility, and interoperability of the enterprise systems [82], [83]. A vital portion of this research revolves around creating and implementing function blocks to enable the traits above. The picture below depicts the generic characteristics present in a contemporary Function block. Individual components of the function block characteristic is described in the forthcoming sections of this thesis.

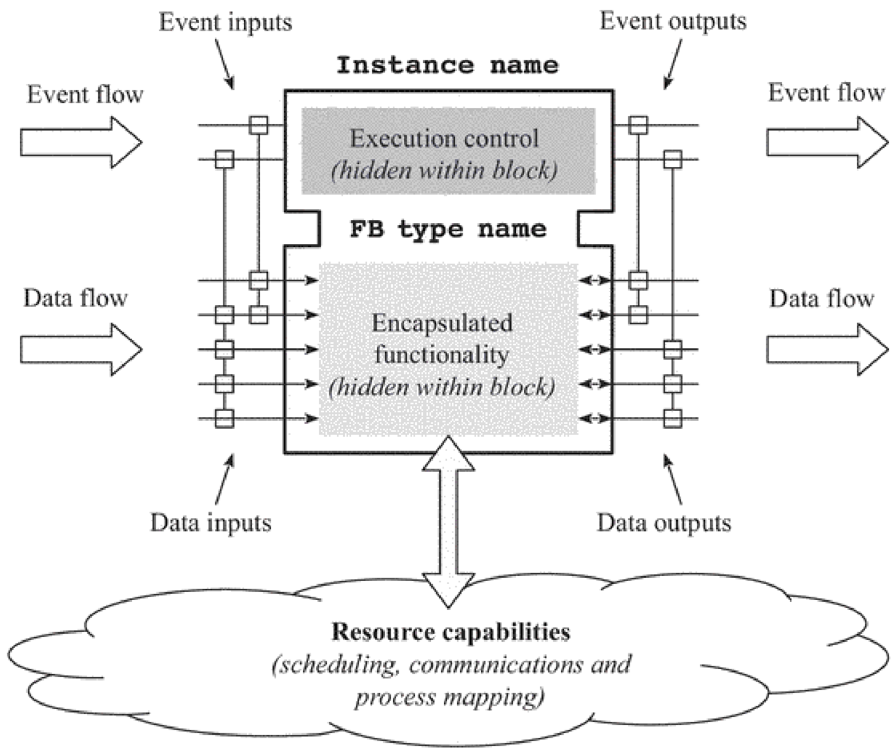


Figure 11. Function block characteristics [82]

Adapted from its predecessor IEC 61131-3 [84], IEC 61499 standard for function block, offers a reference architecture for event-driven execution model and flexible control systems. Authors Strasser et al. [85], establishes a stable open framework environment to create a next-generation distributed and flexible automation environment using the IEC61499 standard which targets portability, configurability, and interoperability [86]. This standard also enables function blocks to implement runtime reconfigurability and scheduling of algorithms during execution between function block networks [87].

2.5 PLANTCockpit OS

Production Logistics and Sustainability Cockpit (PLANTCockpit) [88] provides a flexible integration framework that aims to reduce the complexity of interconnecting various enterprise systems and services across the whole value chain. PLANTCockpit OS is an open source platform deployed over a distributed environment, namely an Enterprise Service Bus (ESB). PLANTCockpit OS provides the necessary configuration modules to reduce integration complexity and construct a consolidated foundation for service-oriented architecture (SOA) [89]. Built on top of well-known standards, PLANTCockpit OS empowers existing and well-established frameworks such as Spring, Apache Karaf, Apache Camel and ActiveMQ. It adheres to well-known standards such as JMS, XML, XSLT, HTML5, SVG and OSGi, IEC 61499. PLANTCockpit OS is platform independent and allows the creation of custom component. PLANTCockpit OS integrates data from diverse sources and provides advanced analytics.

2.5.1 PLANTCockpit – OS in system integration

PLANTCockpit OS is a simple yet powerful system that can function as an integral part of the enterprise architecture which provides a central environment that enables continuous monitoring and optimized controlling of all production and intra-logistical processes. PLANTCockpit OS is a comprehensive framework that proposes to break all enterprise integration barriers and makes business more efficient and sustainable [90].

Dennert et al. [91], implements the SOA paradigm within a Function Block approach using the PLANTCockpit project to observe that service-oriented solution, because of their loose coupling functionality, facilitate flexible & extensible data integration in heterogeneous manufacturing environments. Also [92] and [93] presents an approach handled to integrate the enterprise applications with shop-floor devices using PLANTCockpit platform thus proving that PLANTCockpit is also interoperable.

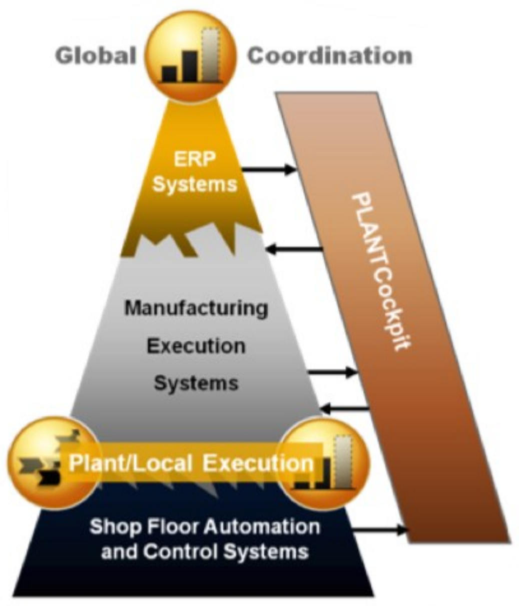

Global
Coordination
ERP Systems
PLANTCockpit
Manufacturing Execution Systems
Plant/Local Execution
Shop Floor Automation and Control Systems

Figure 12. PLANTCockpit in the Automation Pyramid [94]

The ideology of PLANTCockpit and its place in the enterprise can be observed in the figure above. An Automation pyramid is more often represented as a benchmark for defining a common platform for describing various levels of automation and distributed systems in a factory. PLANTCockpit functions as an extension to the enterprise automation pyramid, managing the overall data, communication and information flow across all the levels of enterprise automation pyramid and provides comprehensive visualization of those above. The figure below visualizes the position of PLANTCockpit OS and its interaction with the automation pyramid as defined in IEC 62264 [95], which is also an international standard conceived to promote enterprise-control system integration. All the layers of the pyramid, including the enterprise layer (ERP), manufacturing execution layer (MES) and process control layer acts as data sources to PLANTCockpit.

2.5.2 PLANTCockpit OS in distributed automation

Modern Manufacturing industries demand consolidated and unified view and control over their production environments across all the layers of the supply chain. PLANTCockpit serves this purpose by providing insights of different information systems, flexible data collection & data analysis. On top of the seamless system integration functionalities, PLANTCockpit also provides a visualization layer for visualizing the integrated data and the overall process. One of the critical concepts of PLANTCockpit architecture is to enable modular distributed control systems and provide extensive visibility to monitor and analyze intra-logistical enterprise communications and work flow [90].

Taking into consideration the scope of IEC-61499, this standard was designed as architecture that provides terminology, concepts, and model for implementation of distributed systems [82]. To render the purpose of modularity, reusability, and reconfigurability, PLANTCockpit OS employs Function Block model which have software components and functionalities embedded in them and executed during runtime.

2.6 Learnings and Remarks

Data collected from various systems are generally integrated with a centralized system which allows the manufacturers to continuously monitor and investigate on their process data and ultimately enhance their productivity and profitability. Integration of multiple information systems focuses on integrating different data systems, which provides the users with the feature to interact with one single information system [96]. The risks should also be considered and the problems that Data collection framework should overcome, while collecting data from Legacy Systems should also be addressed. Based on general research and personal study, few of the problems that might arise during data collection are addressed as follows:

- The same data may be collected twice, i.e., collection of duplicate data.
- During Data Collection, Irrelevant data may be collected.
- Continuous data collection sometimes leads to erroneous data being collected
- Sometimes misinterpreted data are also collected.
- Sometimes Database might be disorganized which leads to faulty data collection.
- Taking into consideration the high volumes of data, handling and maintaining the quality of data becomes cumbersome.
- Lack of validation while converting the format of the data or complex transformation of the content of the data.
- When data being collected is from heterogeneous sources, a massive amount of data might be gathered, which will be challenging to manage.
- The data collected from various sources should be secured.
- Semantic Integration of all the data collected.

- The complete Integration chain should be managed Holistically.

Continuous Real-Time Data Collection also comes with problems of its own. They can lead to a variety of issues while trying to integrate those data collected from different information systems [97]:

- While managing large volumes of data from various sources, it might prove to be cumbersome to examine the exact origin or cause of the data or difficult track how the data is being utilized.
- When a data is being changed from the source, it is difficult to be automatically detecting the changes.
- Data collected from various sources may be of many formats, integrating those formats might be a big problem. The implemented system must be compatible with all the available formats. If an additional format is included in the system, the system or database should be modified in such a way that it accepts and adapts to the format.

According to the author in [97], Data Integration problems can be addressed by adapting Process Orchestration along with the ability to abstract the services. The data has to be routed to and from the appropriate systems efficiently in real-time. For this purpose, several adapters are used, and these adapters should also be managed efficiently. Effective management of Semantic and metadata of different information systems can be considered another method to reduce Data Integration problems.

Integration needs to be done on different layers of Automation Pyramid between collaborating systems [92]. Taking into accordance with the interfaces characterized in IEC 62264 – a combination of execution between OPC and ISA has been defined lately, in which exchange of data between different layers is possible. On the other hand, only exchanging data is not sufficient, all the components in each level of the pyramid will interact with each other, which will be a hurdle for consistent integration [92].

The Data Collection Framework Architecture for C2NET should be designed in such a way that

- It should possess the capability to handle the massive amount of data
- Sometimes while collecting data from the legacy systems, the data might not be consistent. DCF should have the ability to solve inaccuracies in the data collected [98].
- The DCF should be able to manage and handle data effectively and efficiently.

All the layers of the Pyramid are based on the different concept of data processing. One of the primary challenges in data integration is bringing together all these layers by following a holistic approach. Another method of using Data Collection Framework is by using IEC61499, which provides the facility to use Function Blocks as a service interface to adapt data from different systems.

3. APPROACH

This section speaks about the data sources and provides deep insights about function block approach. Also, this section describes the software framework and tools adopted to implement the solution. This section also presents the overview/description of the use case which is to be implemented in the next section. This section concludes by presenting a summary of the steps followed to achieve the function block solution.

3.1 Examination of Legacy Systems and data sources

The current generation of manufacturing industries is majorly going through data-driven revolution [99]. The competence of collecting data is critical for an organization to gain a competitive advantage over their adversaries. Given the shortcomings of the current technologies and tools, there is a vast open door for improvements and making the data collection techniques more versatile and effective.

The legacy data are mostly generated within the information systems of the manufacturing industry. The data can be made available from virtually everywhere. The most common data sources of legacy systems in the manufacturing industries include databases, spreadsheets, flat files, ERP, CRM systems and in the recent times, internet of things paradigm have attracted a great deal of attention and is exponentially rising to be a major player in data collection. These interdependent systems that provide connected value chain genuinely rely on transparency and collaboration to ensure a quality product is delivered [100].

The most common type of legacy systems in the enterprise manufacturing domain fall under two categories:

- *Internal Systems:* are custom made enterprise systems either designed by the internal resources or provided by vendors.
- *External Systems:* Third party applications or services provided by external service provider.

Almost all the internal and external systems are classified and characterized under few streamlined legacy systems: ERP, MES, CRM and factory floor systems (example: SCADA, etc.). These systems are spread throughout the entire value chain of the Manufacturing enterprises, and they compose the different layers in the Automation Pyramid. All the hierarchical levels as represented in the Automation Pyramid can be seamlessly integrated using the ISA-95 standards which enables the enterprise and control systems integration.

Manufacturing enterprises nowadays are tended to push towards using multiple heterogeneous legacy systems to meet the overgrowing demand in the market. Implementation of various data sources inevitably prompts to inflation of disparate quantity of data, all of which should be managed efficiently and utilized. Distributed Control Application Platform (DCAP) framework as described in [101] enables the industries to collect data from heterogeneous data sources over a control network. The DCAP provides a modular framework to deploy Control Applications for complex manufacturing systems. The DCAP system model offers a consolidated interface which interconnects the components in manufacturing systems is provided below.

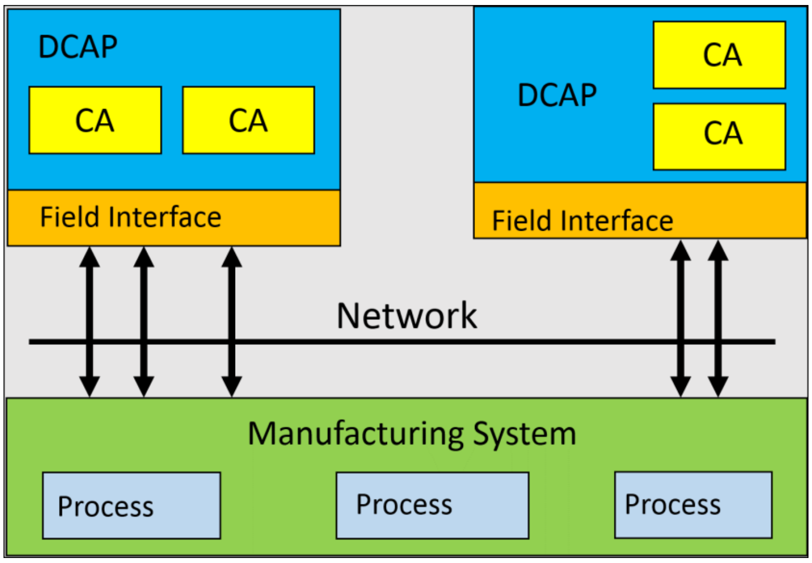


Figure 13. A system model of DCAP[101]

3.2 Adopting Layered Architecture Methodology

The presented PLANTCockpit architecture is demonstrated with five different layers. To prove this research, the majority of the concentration is set upon *"Function Engine layer," "External Systems Layer" and "Presentation Layer."* The figure below exhibits the position and interconnection of the independent layered architecture of the production logistics and sustainability cockpit. These individual and autonomous layers only communicate over uniform interfaces which permits these layers to be designed by different technologies and function independently. Since these layers function independently, their configuration also can be termed independently.

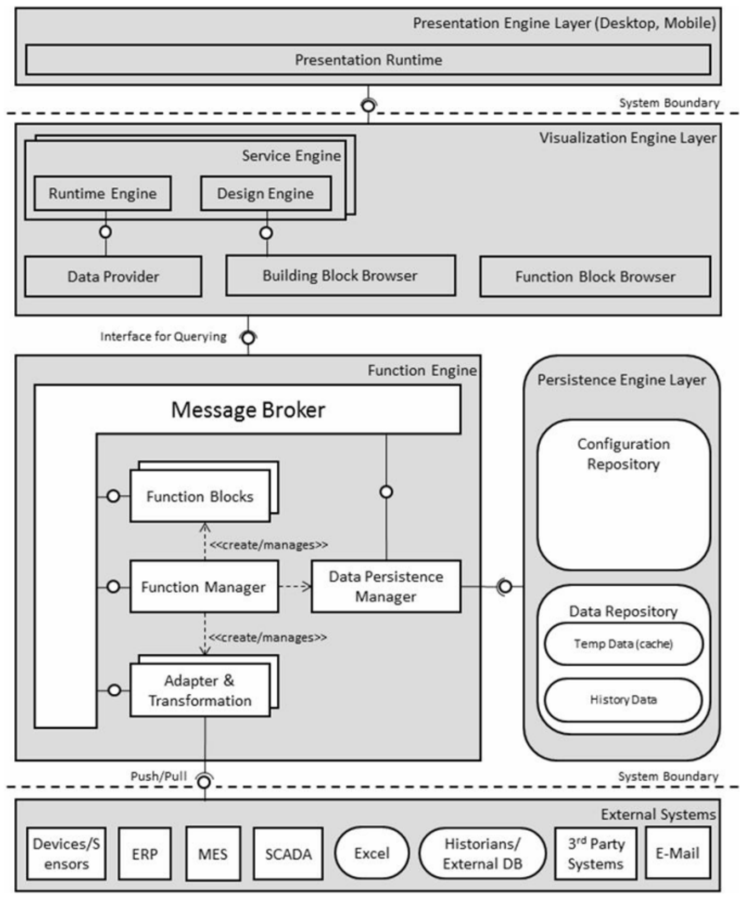


Figure 14. Layered Architecture of PLANTCockpit [90]

3.3 Function Block Approach

The core part of this research approach and implementation revolves around the *Function Engine layer* of the PLANTCockpit architecture. This vital layer provides an extensible execution environment to facilitate the initiation and execution of flexible and interoperable components and functions, in this case, Function blocks. In manufacturing systems, function blocks are considered as a well-acclaimed approach for defining flexible, robust and reusable software components. This well-established concept is utilized to provide an extensive variety of solutions, ranging from a small control valve problem to system integration of an entire production line.

3.3.1 Function Block Structure

Function blocks facilitate industrial algorithms to be encompassed into a single entity that is readily available as well as quickly readable. Every function block comprises of a set of inputs that are interpreted by the internal algorithm during execution which comprises the execution control of the Function Block. Results or the outputs obtained from executing the algorithm is written to the output of the function block. A complete architecture can be structured by implementing a network of function blocks by interconnecting the input and output of individual function blocks, thus generating flexibility in the enterprise solution. The IEC 61499 standard enables modeling and implementating distributed automation systems [102]. The function block is a loosely coupled entity that represents an abstract model of a function that can be implemented as a part of a solution.

3.3.2 Function Block Type

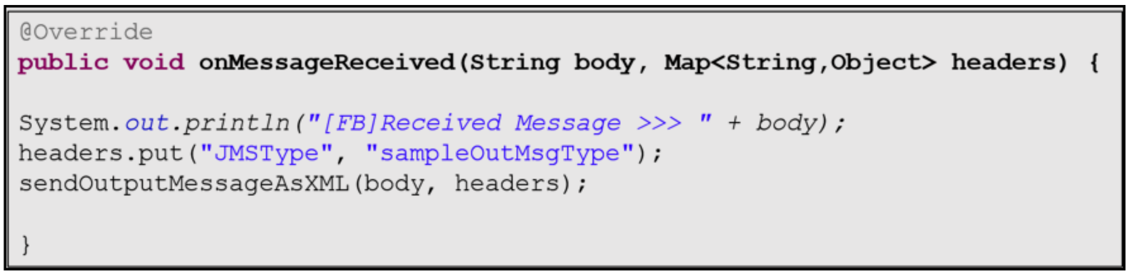
A Function block type is a modular and re-usable entity composed of structural and behavioral components. The Function Manager structures the Function block type. The Function Manager is responsible for overseeing the entire lifecycle of the Function blocks. Starting from the creation of the FB type, initializing the FB instance with respective configuration provisioning, activation of the instance until deactivation and destruction of the FB instance. The position of Function Manager can be witnessed in the PLANTCockpit layered architecture diagram exhibited in Figure 14.

3.3.3 Function Block Instance

As described in section 3.3.2, Function manager is responsible for managing the instance creation of a function block type by injecting them with specific runtime configuration. FB instance themselves, in general, are light-weight components which are nothing but a manifestation of their original FB Type. Being deployed in the Function engine, these lightweight components are executed based on the business configuration during runtime or whenever they are triggered.

3.3.4 Business Logic

The business logic can be loosely termed as the control algorithms or behavior logic which defines the encapsulated functionality of a FB Type. The business logic incorporated in the in the FB Type gets executed during the runtime of the FB instance which is implemented based on event-triggered mechanism. The business logic of all the FB Types, in general, are observed in the “onMessageReceived” function, as this function gets invoked when the instance of the FB Type is triggered.



```
@Override
public void onMessageReceived(String body, Map<String,Object> headers) {

System.out.println("[FB]Received Message >>> " + body);
headers.put("JMSType", "sampleOutMsgType");
sendOutputMessageAsXML(body, headers);

}
```

Figure 15. “onMessageReceived” method snippet

3.3.5 Function Block Network (FBN)

PLANTCockpit OS solution is established upon components that are loosely coupled on a Function Block (FB) scheme. Created adopting the IEC 61499 standard, the FBs offer a standard interface for communication with multiple entities. This interface enables robust combination between the components which favors in solving an individual use case. The communication between the function blocks are ensured by dynamically establising routes between them which functions as a channel for information exhange. This leads to the flexibility of solution for many problems. The Function Block Networks (FBNs) are deployed and evaluated within the Function engine layer of the PLANTCockpit layered reference architecture.

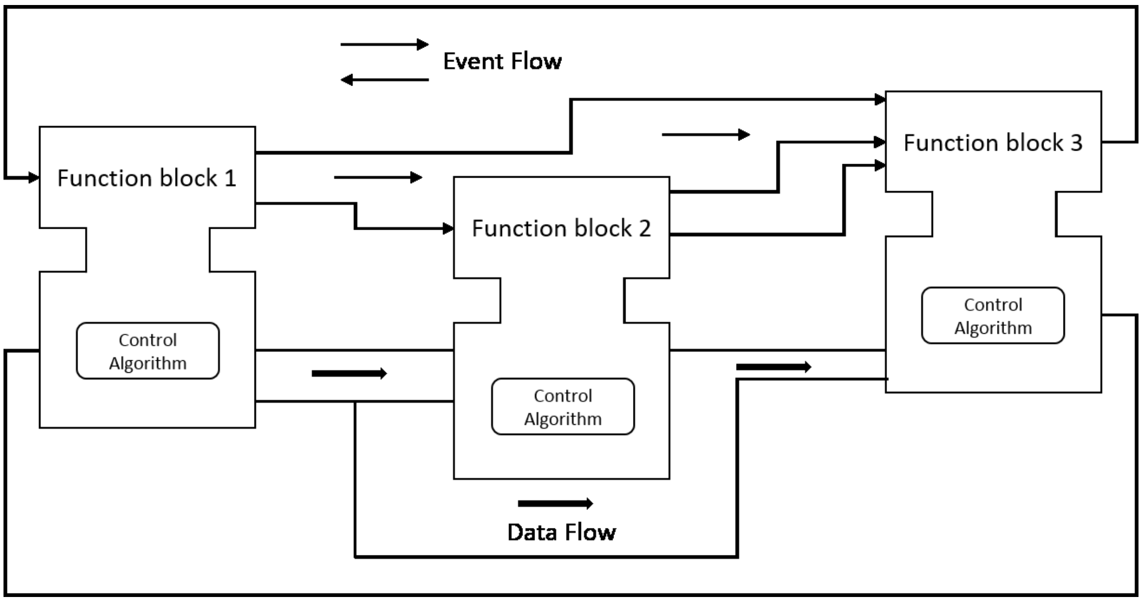


Figure 16. Simple Function block network with data and event flows

3.4 Overview of Software Framework & Tools

3.4.1 Enterprise Service Bus (ESB)

An Enterprise Service Bus (ESB) is a distributed integration middleware that unifies the services encapsulated by the software components, messaging and exchange of information and transformation and routing functionalities which enables consolidating integration of enterprise systems. This software architectural framework enables for designing and implementation of seamless communication between loosely coupled entities and dynamically implement Service-Oriented-Architecture functionalities. The heterogeneous components can be independently deployed disparately over a network.

The full-featured functionality of an ESB depends on the precise structuring of the enterprise message model and proper functionality design of the applications. The ESB solution adopted in this PLANTCockpit deployment is *Apache ServiceMix*.

Apache ServiceMix: Apache ServiceMix[103] is a state of the art distributed ESB. This open-source, flexible and lightweight integration container blends the features and functionalities of diverse tools and provides a stand-alone, powerful runtime platform that enables asynchronous and synchronous messaging protocol between the software components and services deployed thus facilitating the engineering of integration systems. Service Mix is a Java-based enterprise-class ESB solution exclusively powered by Open Service Gateway Initiative (OSGi) bundles. The comprehensive Apache ServiceMix solution constitutes successful functioning and execution of several individual entities such as *Apache Karaf, Apache ActiveMQ, Apache CXF and Apache Camel*. The characteristic features of Apache ServiceMix include:

- OSGi-based server runtime powered by **Apache Karaf.**
- Reliable messaging and asynchronous communication between distributed processes with a Message Oriented Middleware such as **Apache ActiveMQ.**
- Apache CXF facilitates the building and deploying of Web Services in general including RESTful web services.
- **Apache Camel** takes care of Asynchronous messaging coupled with the routing mechanism.

3.4.2 Message Broker

Message Broker is a publish-subscribe component which enables effective communication between the loosely coupled components through messaging mechanisms. The message broker establishes a secure and trustful connection between the deployed entities and ensures high availability and reliable message delivery. The message broker sits on the core part of the ESB and is an integral component to facilitate enterprise integration[104]. PLANTCockpit employs off-the-shelf ActiveMQ message broker solution to ensure higher interoperability level and comply with the existing standards.

Apache ActiveMQ is a well-defined open source message broker capable of providing asynchronous and reliable communication between distributed applications irrespective of its location and protocol. Written with a Java Message Service(JMS) client, ActiveMQ can foster communication from one or more client or server. ActiveMQ employs a robust network of brokers thus enhancing its flexibility in configuration as well as execution and supports a wide range of protocols. ActiveMQ can facilitate in delivering stable and scalable production environments for distributed applications which primarily depend on messaging. Being platform independent, ActiveMQ can efficiently integrate applications written in different languages.

3.4.3 Message Routing

Apache Camel is employed to interconnect the application modules. Apache Camel works together hand in hand with Apache ActiveMQ to provide active routing mechanism and mediation rules for delivering messages. Camel routing ensures effective delivery of messages and routing of the output of the ActiveMQ topic of the FB instance. The Apache Camel empowers robust interconnection of the instances of the function blocks such as adapters and event processors etc. while connecting to the corresponding ActiveMQ endpoints concerning the configured data types of the message.

3.5 Use Case description

The objective of this use case is to autonomously assign works hours to technicians based on an Order or a Project Requirement. The ultimate goal of this research is to implement a network of interconnected modular entities using the PLANTCockpit solution to enable robust data collection from the legacy systems in the manufacturing industry.

For the implementation of this use-case, four different Function Block types are deployed,

1. A **REST Adapter**, capable of consuming RESTful web services based on the REST Endpoints configured.
2. An **MS Excel Adapter**, which reads the Project information from an external spreadsheet legacy system and sends the query to the second function block and routes the results using Camel routing-engine.
3. A **XML Refactor FB,** capable of extracting the XML elements and enables refactoring of XML Structural data.
4. **SQL Database Adapter**, which retrieves the information from the XML Refactor Function Block and Queries the enterprise database by establishing an active connection concerning the requirements of the user provided in the project information

3.6 Summary

First and foremost, a Maven module is created from the PLANTCockpit archetype, which encapsulates the functionalities and template of the Function block template. The Maven module project is then modified to suit the requirement and then packaged as Data Adapters or event processors etc. A maven project is wrapped so that the code and all the dependencies are compiled and built into a bundle (JAR file).

The bundle is deployed to the "deploy" folder of *Apache ServiceMix*. Karaf identifies the applications and runs the application in the karaf container. The deployed applications, in this case, Function blocks can be located using the <fpid>. The default port is 8181. Then each FB type deployed is individually initialized by ServiceMix and configured to create an active instance based on the Business configuration. A Function Block Network (FBN) is then designed to facilitate information flow and message routing within these function blocks thanks to the message broker ActiveMQ and Apache Camel routing engine. This comprehensive methodology aims to enable Enterprise Integration as a whole by implementing the use case.

3.6.1 PLANTCockpit in C2NET Data Collection Client

This research inspires to demonstrate an approach for integrating the legacy systems framework of an enterprise by incorporating a robust framework which can tap into external data sources, perform real-time data collection and enable the transformation of system-specific data and data logistics. In the core of C2NET architecture lies the Data Collection Client (DCC) components which are responsible for effective provisioning and functioning of data collection components. The DCC gathers information from heterogeneous legacy systems and also from the shop-floor IoT devices and transform the data to enable seamless information flow between the interconnected systems. The C2NET DCC is encompassed of two unique components *"Legacy System Hub"* and *"IoT Hub."*

PLANTCockpit framework resides inside the *"Legacy System Hub"* of the C2NET Data Collection Client component making it as a robust interface to connect to enterprise legacy systems for managing information of enterprise operations. PLANTCockpit OS serves as an efficient interface to collect and transform and send data across all layers of the automation pyramid. The core components which interact with C2NET DCC are shown in the figure below.

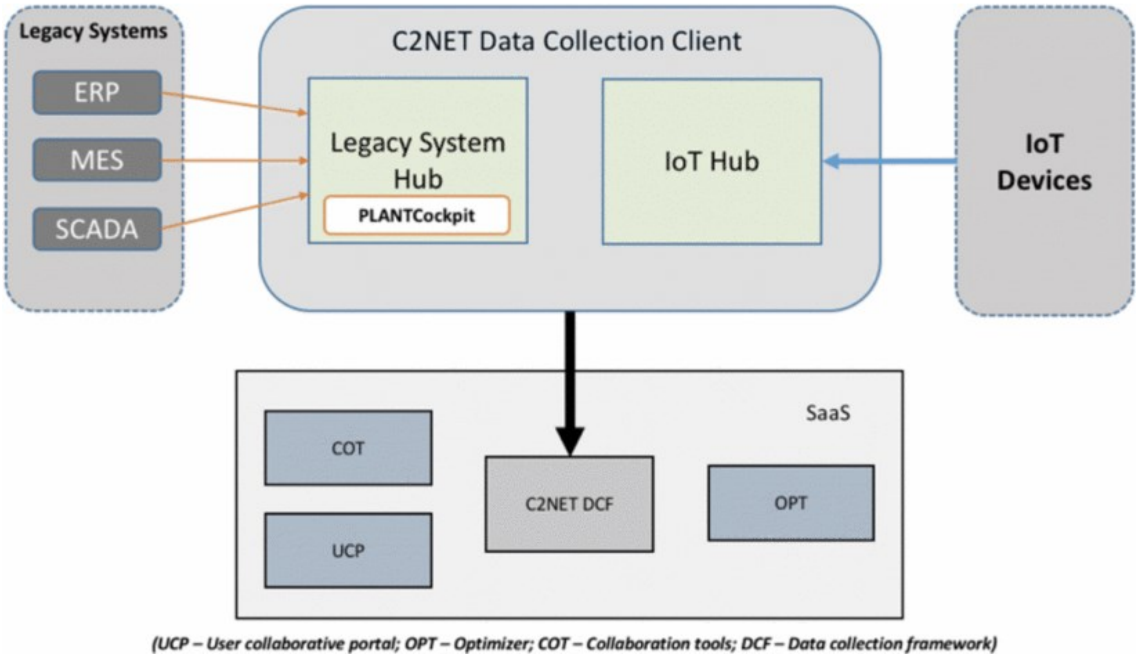

Legacy Systems
ERP
MES
SCADA
C2NET Data Collection Client
Legacy System Hub
PLANTCockpit
IoT Hub
IoT Devices
SaaS
COT
UCP
C2NET DCF
OPT
(UCP – User collaborative portal; OPT – Optimizer; COT – Collaboration tools; DCF – Data collection framework)

Figure 17. Place of PLANTCockpit in the C2NET DCC Reference Architecture [11]

4. IMPLEMENTATION

Chapter 4 describes the business logic of individual Function blocks created and demonstrates the deployment procedure of the Function blocks created using the PLANTCockpit framework. This section illustrates the implementation of the use case and help visualize how enterprise integration in achieved using the proposed solution.

4.1 Legacy system integration using function blocks

This sub-section will provide a detailed analysis of the individual function blocks used to prove the approach by implementing a use case. For this research, four unique function blocks are utilized, that have services encapsulated within them, which are programmed to function independently during the execution or when they are triggered.

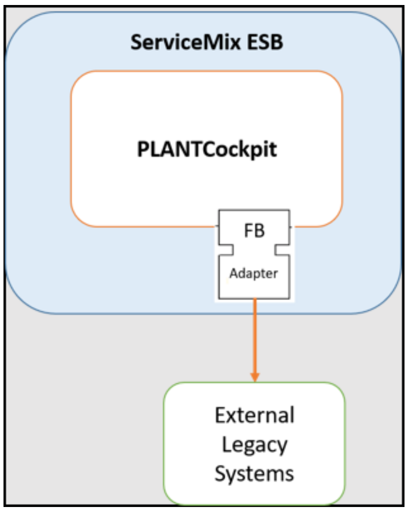


Figure 18. FB Adapter connection to Legacy Systems [11]

This thesis research involves designing and implementation of Function Block Adapters and incorporating with the PLANTCockpit OS solution for acknowledging and permitting the communication between the external legacy systems of the enterprise. This integration is made feasible by effective deployment and utilization of the Data Adapter components which establishes a robust interface between the PLANTCockpit OS solution and heterogeneous legacy systems. The figure above portrays the position of Data Adapter FB component in the ESB component. This setup is deployed in the distributed ESB environment to make this approach probable.

4.1.1 REST Adapter

The Representational State Transfer (REST) adapter is characterized by interchange communication between different remote clients and web services along with the Integration server. REST, in general, is a framework designed for distributed applications that work on HTTP protocol. This highly self-contained application can be invoked from anywhere on the internet thanks to their highly scalable and high flexible architecture.

REST adapter is configured to consume restful web services and enables communication between heterogeneous systems using standard protocols. The REST adapter is programmed to function synchronously and can be exposed to different HTTP endpoints. For this use case implementation, REST adapter provisioned is capable of consuming POST and GET methods. Along with dynamic endpoint definition, the REST adapter provides the feasibility for defining the user's custom attributes.

4.1.2 MS Excel Function block

The MS Excel function block is an adapter which is capable of parsing or reading spreadsheet files that are stored either on the on-premises systems or the cloud. The spreadsheet content extractions are made accessible to the function block using the Apache POI library[105].

The MS Excel adapter is a function block type designed to parse a spreadsheet file which contains production schedule. This FB will be initialized and triggered by the message broker when it receives an order from the REST Adapter.

4.1.3 XML Refactor Function block

This FB acts as a gateway between the SQL connector and other external client applications. The XML refactoring FB takes care of the XML schema structural management changes for us. This FB is capable of restructuring the XML schema to accommodate the requirement which is readable by the next in line SQL connector and send the output to SQL connector for querying. This FB enables custom refactoring of the XML data to adapt to the use case presented.

4.1.4 SQL Function block

The SQL connector extends the functionality of Accessing the database and processing the data by executing the SQL statement injected while configuring the instance of the SQL FB Type. The results can be obtained in custom formats (for example, JSON and XML) and utilized as per the business process.

The SQL FB is programmed, virtually to access all DB that uses SQL query. For this research purposes, MSSQL is taken into consideration, but various relational databases such as MySQL and also PostgreSQL can be used. The SQL FB can be used to query all the RDBMS related DB by taking advantage of the standard JDBC drivers provided by the individual vendors.

4.1.5 Function block Engine Configurator (FBEC)

PLANTCockpit OS also offers a user-friendly graphical system that serves as an efficient system configuration tool. FBEC provides a holistic GUI that enables the user to simplify the orchestration and usability of Function block networks (FBNs) to serve a custom scenario or a use case.

FBEC provides a comprehensive and intuitive, yet simple web-based configuration user interface tool that facilitates secure administration and monitoring of the entire system from anywhere around the world. FBEC is characterized to reduce the complexity of configuration and increases the transparency of heterogeneous system integration landscape.

4.2 Use case implementation

The sub-section illustrates a better picture by representing the described approach in a use case. By the end of this section, a right ideology can be inferred as to how PLANTCockpit OS would fit in the architecture of C2NET Legacy System Hub.

The FB instance configuration and execution can be done in 2 environments. One is using the barebone Apache Active MQ message broker and initializing the instances and defining the routes manually in camel route manager.

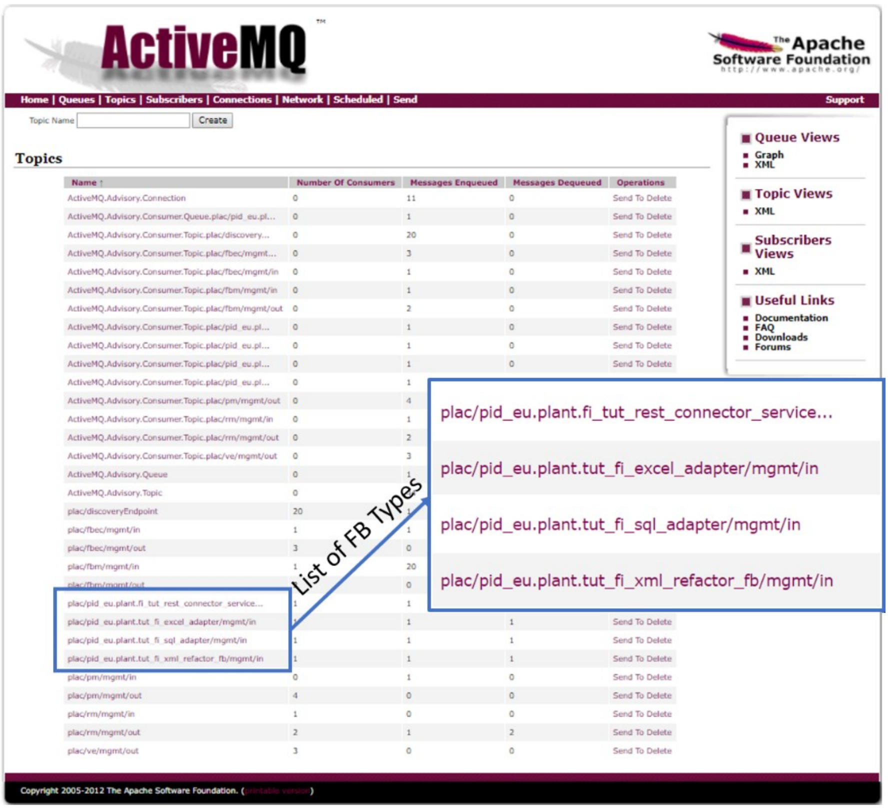
The second one is using FBEC. Let FBEC do all the work for you. It is a GUI that acts as a visualization layer built on top of the message broker and route manager and provides optimized configuration platform for ease of use.

The first step in implementing the FB networks which includes configuration and creating the routes between them (either by ActiveMQ or by FBEC GUI), is to initialize the ESB where the FBs reside which automatically triggers the initialization of the FBs and make them active. Once the ESB, in this case, Apache ServiceMix is started, All the FBs which are compiled as Java executable OSGi bundles are initiated.

4.2.1 FB Type Instance Configuration

4.2.1.1 Instance configuration using ActiveMQ

Once the ServiceMix container is initialized, the web-based ActiveMQ message broker library that has been installed becomes active. The initialized FBs appear as topics on the ActiveMQ page. The ActiveMQ web-page is invoked for creating an instance of FB Type and defining routes between the FBs. Since a FBN is being implemented, all the instances are formed under the Function Block manager and provide the identifier name of the function block being initialized. Below is the representation of how each function blocks are initialized. The initialized FB Types is found on the "Topics" section of the ActiveMQ, message broker. The ActiveMQ web page as displayed in the picture below can be viewed by accessing the URL *http://localhost:8181/activemqweb/topics.jsp*.


ActiveMQ™
The Apache Software Foundation
http://www.apache.org/
Home | Queues | Topics | Subscribers | Connections | Network | Scheduled | Send
Support
Topic Name
Create
Queue Views
Graph
XML
Topic Views
XML
Subscribers Views
XML
Useful Links
Documentation
FAQ
Downloads
Forums
List of FB Types
plac/pid_eu.plant.fi_tut_rest_connector_service...
plac/pid_eu.plant.tut_fi_excel_adapter/mgmt/in
plac/pid_eu.plant.tut_fi_sql_adapter/mgmt/in
plac/pid_eu.plant.tut_fi_xml_refactor_fb/mgmt/in

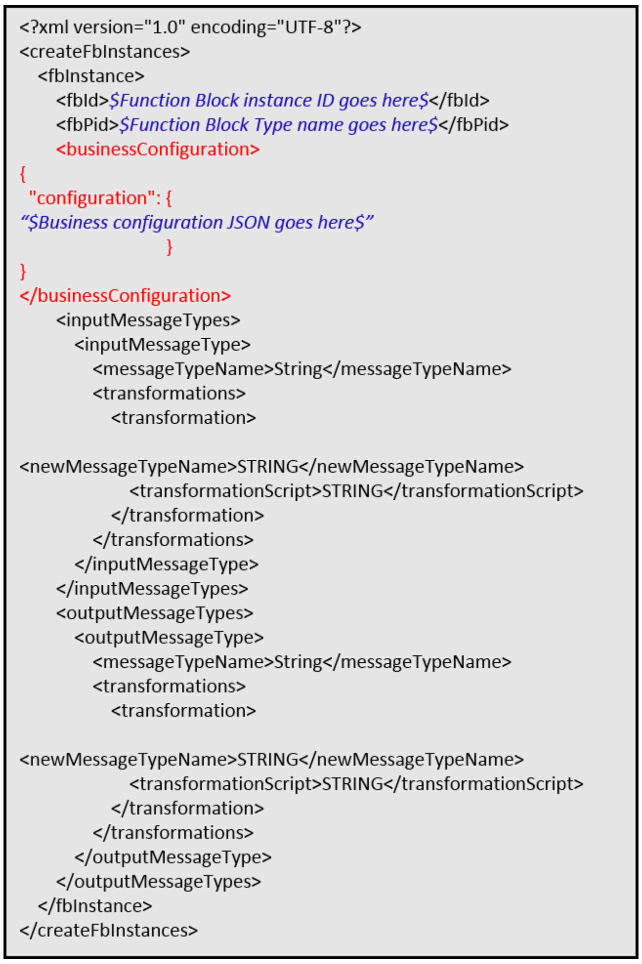
Topics

Name ↑	Number Of Consumers	Messages Enqueued	Messages Dequeued	Operations
ActiveMQ.Advisory.Connection	0	11	0	Send To Delete
ActiveMQ.Advisory.Consumer.Queue.plac/pid_eu.pl...	0	1	0	Send To Delete
ActiveMQ.Advisory.Consumer.Topic.plac/discovery...	0	20	0	Send To Delete
ActiveMQ.Advisory.Consumer.Topic.plac/fbec/mgmt...	0	3	0	Send To Delete
ActiveMQ.Advisory.Consumer.Topic.plac/fbec/mgmt/in	0	1	0	Send To Delete
ActiveMQ.Advisory.Consumer.Topic.plac/fbm/mgmt/in	0	1	0	Send To Delete
ActiveMQ.Advisory.Consumer.Topic.plac/fbm/mgmt/out	0	2	0	Send To Delete
ActiveMQ.Advisory.Consumer.Topic.plac/pid_eu.pl...	0	1	0	Send To Delete
ActiveMQ.Advisory.Consumer.Topic.plac/pid_eu.pl...	0	1	0	Send To Delete
ActiveMQ.Advisory.Consumer.Topic.plac/pid_eu.pl...	0	1	0	Send To Delete
ActiveMQ.Advisory.Consumer.Topic.plac/pid_eu.pl...	0	1		
ActiveMQ.Advisory.Consumer.Topic.plac/pm/mgmt/out	0	4		
ActiveMQ.Advisory.Consumer.Topic.plac/rm/mgmt/in	0	1		
ActiveMQ.Advisory.Consumer.Topic.plac/rm/mgmt/out	0	2		
ActiveMQ.Advisory.Consumer.Topic.plac/ve/mgmt/out	0	3		
ActiveMQ.Advisory.Queue	0	1		
ActiveMQ.Advisory.Topic	0	[illegible]		
plac/discoveryEndpoint	20	1		
plac/fbec/mgmt/in	1	1		
plac/fbec/mgmt/out	3	0		
plac/fbm/mgmt/in	1	20		
plac/fbm/mgmt/out	[illegible]	0		
plac/pid_eu.plant.fi_tut_rest_connector_service...	1	1		
plac/pid_eu.plant.tut_fi_excel_adapter/mgmt/in	1	1	1	Send To Delete
plac/pid_eu.plant.tut_fi_sql_adapter/mgmt/in	1	1	1	Send To Delete
plac/pid_eu.plant.tut_fi_xml_refactor_fb/mgmt/in	1	1	1	Send To Delete
plac/pm/mgmt/in	0	1	0	Send To Delete
plac/pm/mgmt/out	4	0	0	Send To Delete
plac/rm/mgmt/in	1	0	0	Send To Delete
plac/rm/mgmt/out	2	1	2	Send To Delete
plac/ve/mgmt/out	3	0	0	Send To Delete

Copyright 2005-2012 The Apache Software Foundation. (printable version)

Figure 19. Function Block Types in Apache ActiveMQ

A vital part of creating an instance is the use of Business configuration(BC) message. The structure of the BC varies depending on the FB Type. The instances of each FB Type are initiated by sending the Business configuration(BC) as a JMS message to the ServiceMix container. The actual business configuration in the business configuration message are characterized in a JSON format. A generic BC template is represented below.



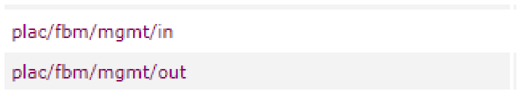
```
<?xml version="1.0" encoding="UTF-8"?>
<createFbInstances>
  <fbInstance>
    <fbId>$Function Block instance ID goes here$</fbId>
    <fbPid>$Function Block Type name goes here$</fbPid>
    <businessConfiguration>
{
 "configuration": {
"$Business configuration JSON goes here$"
                }
}
</businessConfiguration>
    <inputMessageTypes>
      <inputMessageType>
        <messageTypeName>String</messageTypeName>
        <transformations>
          <transformation>

<newMessageTypeName>STRING</newMessageTypeName>
            <transformationScript>STRING</transformationScript>
          </transformation>
        </transformations>
      </inputMessageType>
    </inputMessageTypes>
    <outputMessageTypes>
      <outputMessageType>
        <messageTypeName>String</messageTypeName>
        <transformations>
          <transformation>

<newMessageTypeName>STRING</newMessageTypeName>
            <transformationScript>STRING</transformationScript>
          </transformation>
        </transformations>
      </outputMessageType>
    </outputMessageTypes>
  </fbInstance>
</createFbInstances>
```

Figure 20. Business Configuration template

The Function block manager can be accessed using the topic *"plac/fbm/mgmt/in."* As the name suggests, the function block manager is utilized to configure individual FB Types and initiate the FB Type instance and manages the entire function block network.



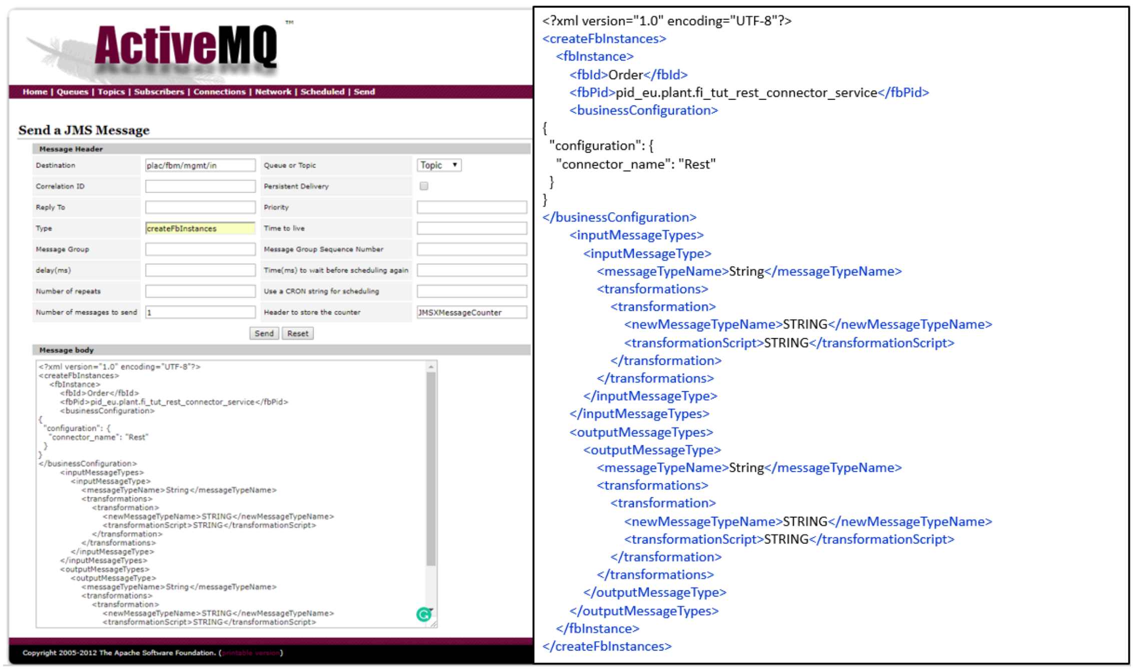
plac/fbm/mgmt/in

plac/fbm/mgmt/out

Figure 21. ActiveMQ Function Block manager topic

4.2.1.1.1 REST Adapter Configuration

In the ActiveMQ message broker, the REST adapter which is packaged as a dynamic OSGi bundle is initialized first. In this use case, there is no specific runtime configuration for this function block. All the business logic required to function the REST adapter is programmed, integrated and encapsulated as one single entity and deployed. The figure below shows the Business configuration utilized to initiate the REST FB Type.


ActiveMQ
Home | Queues | Topics | Subscribers | Connections | Network | Scheduled | Send

Send a JMS Message

Message Header

Destination	plac/fbm/mgmt/in	Queue or Topic	Topic
Correlation ID		Persistent Delivery	☐
Reply To		Priority	
Type	createFbInstances	Time to live	
Message Group		Message Group Sequence Number	
delay(ms)		Time(ms) to wait before scheduling again	
Number of repeats		Use a CRON string for scheduling	
Number of messages to send	1	Header to store the counter	JMSXMessageCounter

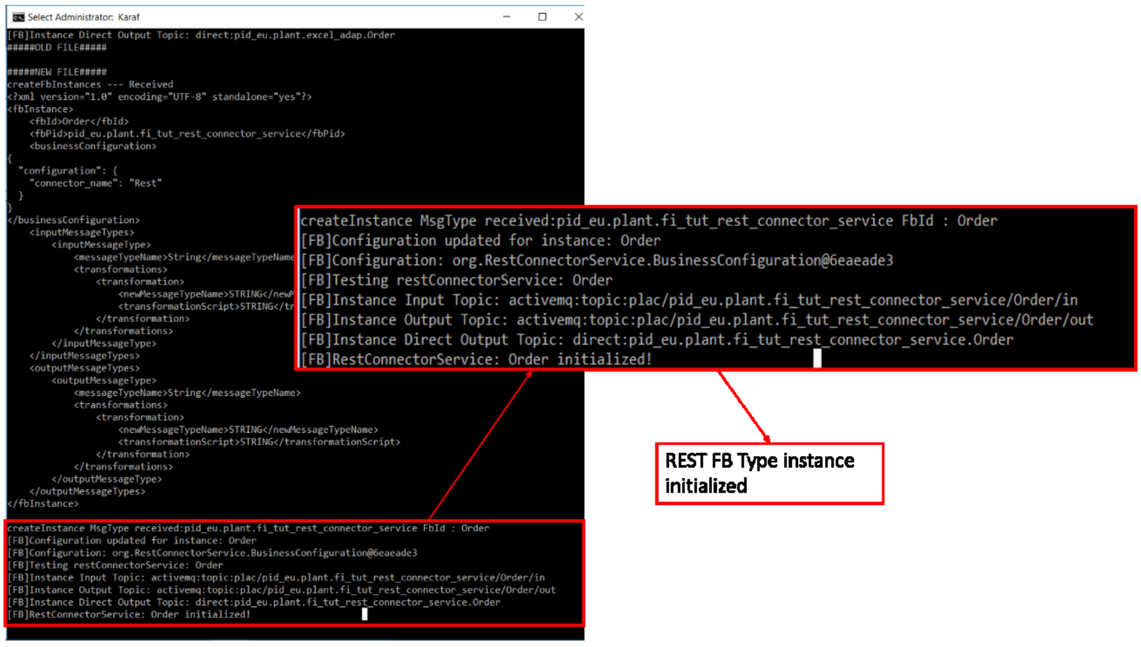
Message body

```
<?xml version="1.0" encoding="UTF-8"?>
<createFbInstances>
   <fbInstance>
      <fbId>Order</fbId>
      <fbPid>pid_eu.plant.fi_tut_rest_connector_service</fbPid>
      <businessConfiguration>
{
 "configuration": {
  "connector_name": "Rest"
 }
}
</businessConfiguration>
      <inputMessageTypes>
         <inputMessageType>
            <messageTypeName>String</messageTypeName>
            <transformations>
               <transformation>
                  <newMessageTypeName>STRING</newMessageTypeName>
                  <transformationScript>STRING</transformationScript>
               </transformation>
            </transformations>
         </inputMessageType>
      </inputMessageTypes>
      <outputMessageTypes>
         <outputMessageType>
            <messageTypeName>String</messageTypeName>
            <transformations>
               <transformation>
                  <newMessageTypeName>STRING</newMessageTypeName>
                  <transformationScript>STRING</transformationScript>
```

```
<?xml version="1.0" encoding="UTF-8"?>
<createFbInstances>
  <fbInstance>
    <fbId>Order</fbId>
    <fbPid>pid_eu.plant.fi_tut_rest_connector_service</fbPid>
    <businessConfiguration>
{
 "configuration": {
  "connector_name": "Rest"
 }
}
</businessConfiguration>
    <inputMessageTypes>
      <inputMessageType>
        <messageTypeName>String</messageTypeName>
        <transformations>
          <transformation>
            <newMessageTypeName>STRING</newMessageTypeName>
            <transformationScript>STRING</transformationScript>
          </transformation>
        </transformations>
      </inputMessageType>
    </inputMessageTypes>
    <outputMessageTypes>
      <outputMessageType>
        <messageTypeName>String</messageTypeName>
        <transformations>
          <transformation>
            <newMessageTypeName>STRING</newMessageTypeName>
            <transformationScript>STRING</transformationScript>
          </transformation>
        </transformations>
      </outputMessageType>
    </outputMessageTypes>
  </fbInstance>
</createFbInstances>
```

Figure 22. REST FB instance configuration

The output from the shell console indicating the REST FB Type being created is shown below.



```
[FB]Instance Direct Output Topic: direct:pid_eu.plant.excel_adap.Order
#####OLD FILE#####

#####NEW FILE#####
createFbInstances --- Received
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<fbInstance>
    <fbId>Order</fbId>
    <fbPid>pid_eu.plant.fi_tut_rest_connector_service</fbPid>
    <businessConfiguration>
{
  "configuration": {
    "connector_name": "Rest"
  }
}
</businessConfiguration>
    <inputMessageTypes>
        <inputMessageType>
            <messageTypeName>String</messageTypeName
            <transformations>
                <transformation>
                    <newMessageTypeName>STRING</newM
                    <transformationScript>STRING</tr
                </transformation>
            </transformations>
        </inputMessageType>
    </inputMessageTypes>
    <outputMessageTypes>
        <outputMessageType>
            <messageTypeName>String</messageTypeName>
            <transformations>
                <transformation>
                    <newMessageTypeName>STRING</newMessageTypeName>
                    <transformationScript>STRING</transformationScript>
                </transformation>
            </transformations>
        </outputMessageType>
    </outputMessageTypes>
</fbInstance>
createInstance MsgType received:pid_eu.plant.fi_tut_rest_connector_service FbId : Order
[FB]Configuration updated for instance: Order
[FB]Configuration: org.RestConnectorService.BusinessConfiguration@6eaeade3
[FB]Testing restConnectorService: Order
[FB]Instance Input Topic: activemq:topic:plac/pid_eu.plant.fi_tut_rest_connector_service/Order/in
[FB]Instance Output Topic: activemq:topic:plac/pid_eu.plant.fi_tut_rest_connector_service/Order/out
[FB]Instance Direct Output Topic: direct:pid_eu.plant.fi_tut_rest_connector_service.Order
[FB]RestConnectorService: Order initialized!
```

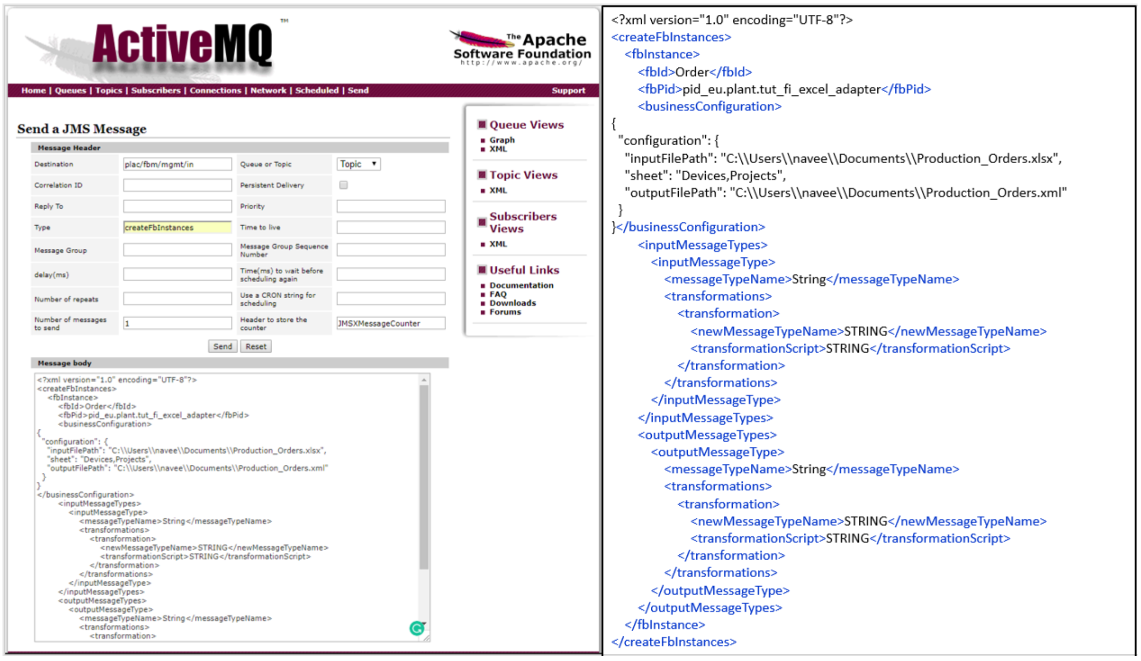
```
createInstance MsgType received:pid_eu.plant.fi_tut_rest_connector_service FbId : Order
[FB]Configuration updated for instance: Order
[FB]Configuration: org.RestConnectorService.BusinessConfiguration@6eaeade3
[FB]Testing restConnectorService: Order
[FB]Instance Input Topic: activemq:topic:plac/pid_eu.plant.fi_tut_rest_connector_service/Order/in
[FB]Instance Output Topic: activemq:topic:plac/pid_eu.plant.fi_tut_rest_connector_service/Order/out
[FB]Instance Direct Output Topic: direct:pid_eu.plant.fi_tut_rest_connector_service.Order
[FB]RestConnectorService: Order initialized!
```

Figure 23. REST FB instance initialized console output

4.2.1.1.2 *Excel Adapter Configuration*

Excel adapter FB Type requires specific parameters that need to be provided while creating the instance. The Excel adapter needs the information such as the

- “Input File location” of the MS Excel file being parsed for that FB Type instance. The location specified can be either be in the local directory or the cloud server.
- “Sheet name” to be considered while parsing elements from the MS Excel file. If no sheet name is specified, the FB will parse and extract all the existing sheets in that particular MS Excel file.
- “Output File name and location” field is optional. This field can be utilized if the user wants to save the parsed XML data from the MS Excel file for future purpose.



Send a JMS Message

Message Header

Destination	plac/fbm/mgmt/in	Queue or Topic	Topic
Correlation ID		Persistent Delivery	☐
Reply To		Priority	
Type	createFbInstances	Time to live	
Message Group		Message Group Sequence Number	
delay(ms)		Time(ms) to wait before scheduling again	
Number of repeats		Use a CRON string for scheduling	
Number of messages to send	1	Header to store the counter	JMSXMessageCounter

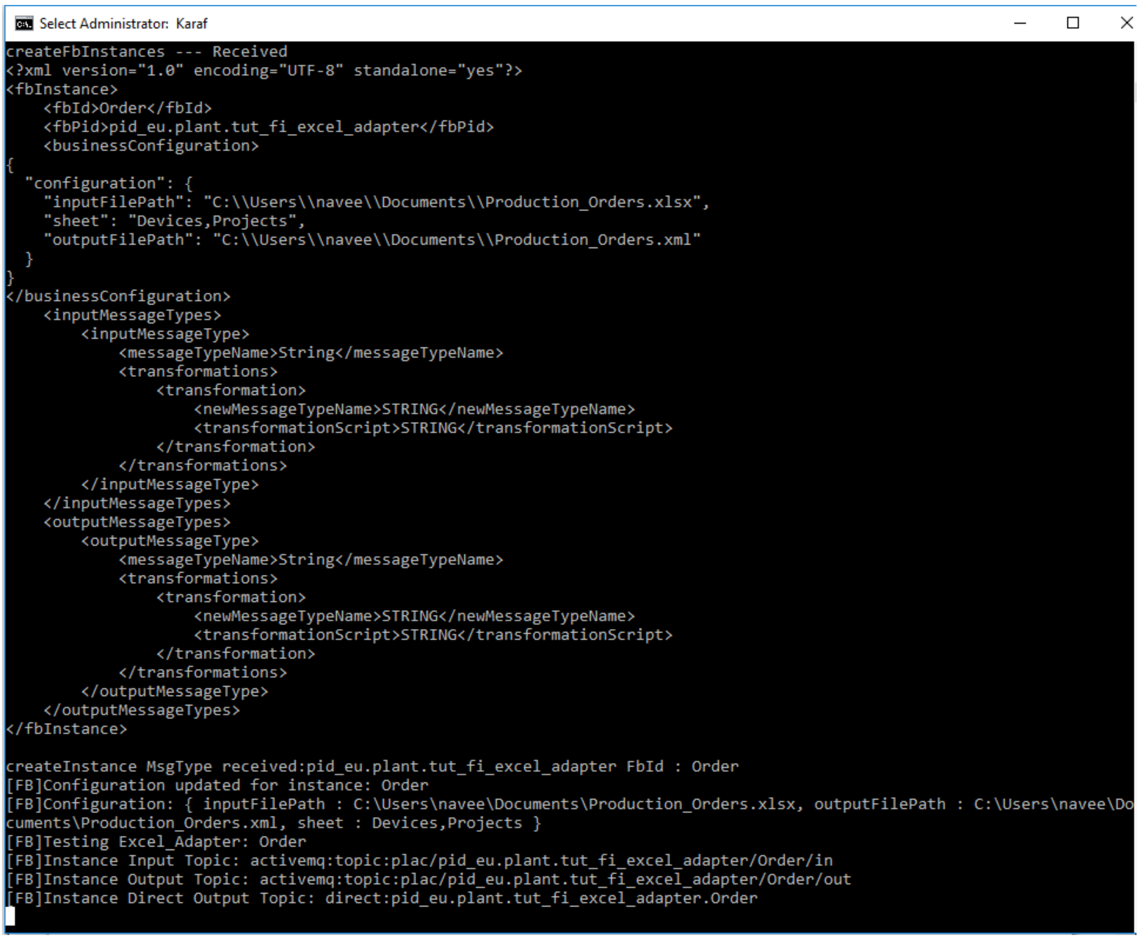
Message body

```
<?xml version="1.0" encoding="UTF-8"?>
<createFbInstances>
  <fbInstance>
    <fbId>Order</fbId>
    <fbPid>pid_eu.plant.tut_fi_excel_adapter</fbPid>
    <businessConfiguration>
{
 "configuration": {
  "inputFilePath": "C:\\Users\\navee\\Documents\\Production_Orders.xlsx",
  "sheet": "Devices,Projects",
  "outputFilePath": "C:\\Users\\navee\\Documents\\Production_Orders.xml"
 }
}
</businessConfiguration>
    <inputMessageTypes>
      <inputMessageType>
        <messageTypeName>String</messageTypeName>
        <transformations>
          <transformation>
            <newMessageTypeName>STRING</newMessageTypeName>
            <transformationScript>STRING</transformationScript>
          </transformation>
        </transformations>
      </inputMessageType>
    </inputMessageTypes>
    <outputMessageTypes>
      <outputMessageType>
        <messageTypeName>String</messageTypeName>
        <transformations>
          <transformation>
```

```
<?xml version="1.0" encoding="UTF-8"?>
<createFbInstances>
  <fbInstance>
    <fbId>Order</fbId>
    <fbPid>pid_eu.plant.tut_fi_excel_adapter</fbPid>
    <businessConfiguration>
{
 "configuration": {
  "inputFilePath": "C:\\Users\\navee\\Documents\\Production_Orders.xlsx",
  "sheet": "Devices,Projects",
  "outputFilePath": "C:\\Users\\navee\\Documents\\Production_Orders.xml"
 }
}</businessConfiguration>
    <inputMessageTypes>
      <inputMessageType>
        <messageTypeName>String</messageTypeName>
        <transformations>
          <transformation>
            <newMessageTypeName>STRING</newMessageTypeName>
            <transformationScript>STRING</transformationScript>
          </transformation>
        </transformations>
      </inputMessageType>
    </inputMessageTypes>
    <outputMessageTypes>
      <outputMessageType>
        <messageTypeName>String</messageTypeName>
        <transformations>
          <transformation>
            <newMessageTypeName>STRING</newMessageTypeName>
            <transformationScript>STRING</transformationScript>
          </transformation>
        </transformations>
      </outputMessageType>
    </outputMessageTypes>
  </fbInstance>
</createFbInstances>
```

Figure 24. Excel FB instance configuration

The corresponding output for the creation of Excel Adapter Type instance can be verified in the ServiceMix console below:



```
createFbInstances --- Received
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<fbInstance>
    <fbId>Order</fbId>
    <fbPid>pid_eu.plant.tut_fi_excel_adapter</fbPid>
    <businessConfiguration>
{
  "configuration": {
    "inputFilePath": "C:\\Users\\navee\\Documents\\Production_Orders.xlsx",
    "sheet": "Devices,Projects",
    "outputFilePath": "C:\\Users\\navee\\Documents\\Production_Orders.xml"
  }
}
</businessConfiguration>
    <inputMessageTypes>
        <inputMessageType>
            <messageTypeName>String</messageTypeName>
            <transformations>
                <transformation>
                    <newMessageTypeName>STRING</newMessageTypeName>
                    <transformationScript>STRING</transformationScript>
                </transformation>
            </transformations>
        </inputMessageType>
    </inputMessageTypes>
    <outputMessageTypes>
        <outputMessageType>
            <messageTypeName>String</messageTypeName>
            <transformations>
                <transformation>
                    <newMessageTypeName>STRING</newMessageTypeName>
                    <transformationScript>STRING</transformationScript>
                </transformation>
            </transformations>
        </outputMessageType>
    </outputMessageTypes>
</fbInstance>

createInstance MsgType received:pid_eu.plant.tut_fi_excel_adapter FbId : Order
[FB]Configuration updated for instance: Order
[FB]Configuration: { inputFilePath : C:\Users\navee\Documents\Production_Orders.xlsx, outputFilePath : C:\Users\navee\Do
cuments\Production_Orders.xml, sheet : Devices,Projects }
[FB]Testing Excel_Adapter: Order
[FB]Instance Input Topic: activemq:topic:plac/pid_eu.plant.tut_fi_excel_adapter/Order/in
[FB]Instance Output Topic: activemq:topic:plac/pid_eu.plant.tut_fi_excel_adapter/Order/out
[FB]Instance Direct Output Topic: direct:pid_eu.plant.tut_fi_excel_adapter.Order
```

Figure 25. Excel FB instance initialized console output

4.2.1.1.3 SQL Adapter Configuration

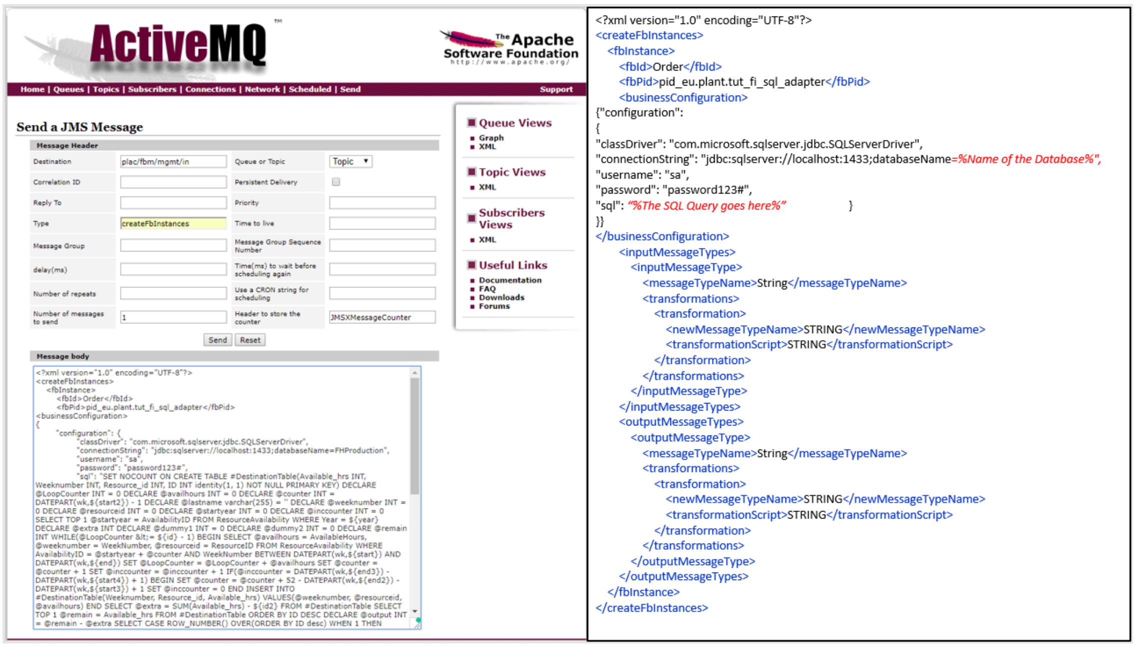
To evaluate the potential of the concept of configuration, the SQL connector FB Type would serve as an excellent example for this use case. This FB Type, when invoked, makes a secured connection to the SQL server based on the configurations provided and queries the DB as per the SQL query specified during the setup.

The mandatory parameters to be provided while configuring an instance of SQL Connector FB Type are:

- "*Username*": Username of the external database to be accessed.
- "*Password*": Password of the external database to be accessed.
- "*Connection String*": This attribute specifies the FB instance the type of the SQL server that needs to be connected. The connection string also includes the database name to be connected to.
- "Class Driver": A JDBC driver that facilitates seamless connection to the SQL database.
- "SQL": The actual SQL query the needs to be executed on the DB, once a successful connection has been stablished.

The output the query is then converted to XML format and saved for future requirement.

The instance configuration of the SQL FB Type and its corresponding console output can be seen in the pictures below. The FB Type instance is created by sending JMS message of the business configuration JSON, incorporated in the *"<createFbInstance>"* XML template. This process is similar to the instance creation of all the FB Types.



Send a JMS Message

Message Header

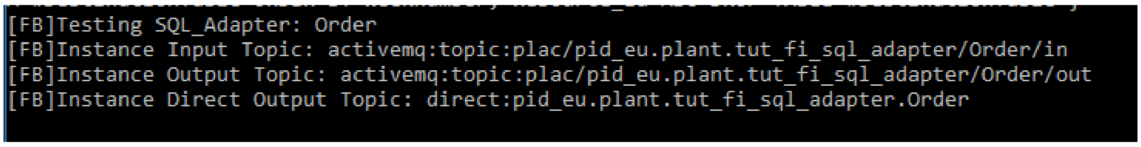
Destination	plac/fbm/mgmt/in	Queue or Topic	Topic
Correlation ID		Persistent Delivery	☐
Reply To		Priority	
Type	createFbInstances	Time to live	
Message Group		Message Group Sequence Number	
delay(ms)		Time(ms) to wait before scheduling again	
Number of repeats		Use a CRON string for scheduling	
Number of messages to send	1	Header to store the counter	JMSXMessageCounter

Message body

```
<?xml version="1.0" encoding="UTF-8"?>
<createFbInstances>
   <fbInstance>
      <fbId>Order</fbId>
      <fbPid>pid_eu.plant.tut_fi_sql_adapter</fbPid>
<businessConfiguration>
{
      "configuration": {
            "classDriver": "com.microsoft.sqlserver.jdbc.SQLServerDriver",
            "connectionString": "jdbc:sqlserver://localhost:1433;databaseName=FHProduction",
            "username": "sa",
            "password": "password123#",
            "sql": "SET NOCOUNT ON CREATE TABLE #DestinationTable(Available_hrs INT,
Weeknumber INT, Resource_id INT, ID INT identity(1, 1) NOT NULL PRIMARY KEY) DECLARE
@LoopCounter INT = 0 DECLARE @availhours INT = 0 DECLARE @counter INT =
DATEPART(wk,${start2}) - 1 DECLARE @lastname varchar(255) = '' DECLARE @weeknumber INT =
0 DECLARE @resourceid INT = 0 DECLARE @startyear INT = 0 DECLARE @inccounter INT = 0
SELECT TOP 1 @startyear = AvailabilityID FROM ResourceAvailability WHERE Year = ${year}
DECLARE @extra INT DECLARE @dummy1 INT = 0 DECLARE @dummy2 INT = 0 DECLARE @remain
INT WHILE(@LoopCounter &lt;= ${id} - 1) BEGIN SELECT @availhours = AvailableHours,
@weeknumber = WeekNumber, @resourceid = ResourceID FROM ResourceAvailability WHERE
AvailabilityID = @startyear + @counter AND WeekNumber BETWEEN DATEPART(wk,${start}) AND
DATEPART(wk,${end}) SET @LoopCounter = @LoopCounter + @availhours SET @counter =
@counter + 1 SET @inccounter = @inccounter + 1 IF(@inccounter = DATEPART(wk,${end3}) -
DATEPART(wk,${start4}) + 1) BEGIN SET @counter = @counter + 52 - DATEPART(wk,${end2}) -
DATEPART(wk,${start3}) + 1 SET @inccounter = 0 END INSERT INTO
#DestinationTable(Weeknumber, Resource_id, Available_hrs) VALUES(@weeknumber, @resourceid,
@availhours) END SELECT @extra = SUM(Available_hrs) - ${id2} FROM #DestinationTable SELECT
TOP 1 @remain = Available_hrs FROM #DestinationTable ORDER BY ID DESC DECLARE @output INT
= @remain - @extra SELECT CASE ROW_NUMBER() OVER(ORDER BY ID desc) WHEN 1 THEN
```

```
<?xml version="1.0" encoding="UTF-8"?>
<createFbInstances>
  <fbInstance>
    <fbId>Order</fbId>
    <fbPid>pid_eu.plant.tut_fi_sql_adapter</fbPid>
    <businessConfiguration>
{"configuration":
{
"classDriver": "com.microsoft.sqlserver.jdbc.SQLServerDriver",
"connectionString": "jdbc:sqlserver://localhost:1433;databaseName=%Name of the Database%",
"username": "sa",
"password": "password123#",
"sql": "%The SQL Query goes here%"              }
}}
</businessConfiguration>
    <inputMessageTypes>
      <inputMessageType>
        <messageTypeName>String</messageTypeName>
        <transformations>
          <transformation>
            <newMessageTypeName>STRING</newMessageTypeName>
            <transformationScript>STRING</transformationScript>
          </transformation>
        </transformations>
      </inputMessageType>
    </inputMessageTypes>
    <outputMessageTypes>
      <outputMessageType>
        <messageTypeName>String</messageTypeName>
        <transformations>
          <transformation>
            <newMessageTypeName>STRING</newMessageTypeName>
            <transformationScript>STRING</transformationScript>
          </transformation>
        </transformations>
      </outputMessageType>
    </outputMessageTypes>
  </fbInstance>
</createFbInstances>
```

Figure 26. SQL FB instance configuration

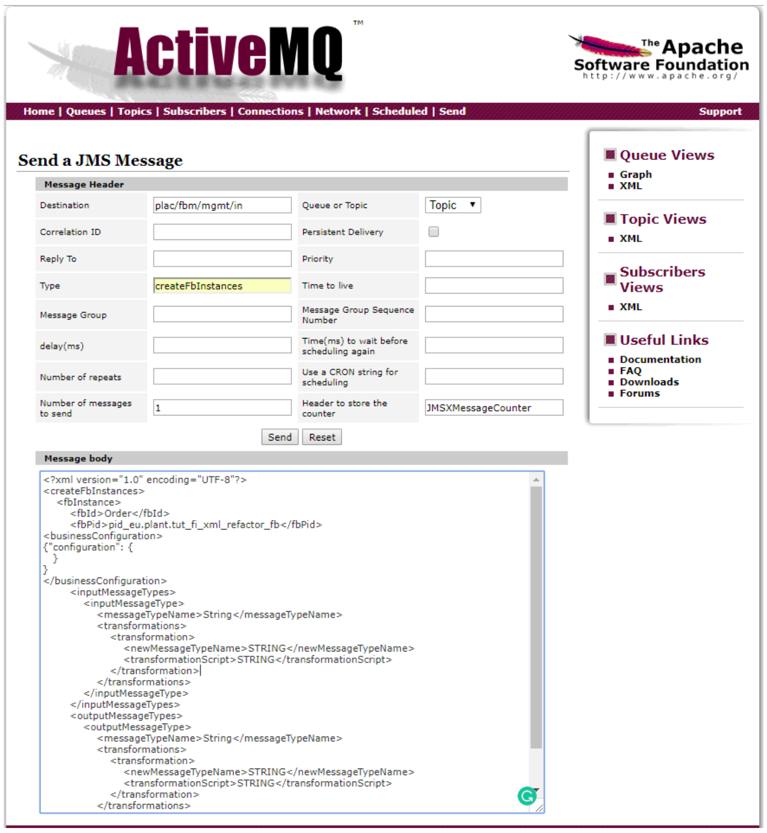


```
[FB]Testing SQL_Adapter: Order
[FB]Instance Input Topic: activemq:topic:plac/pid_eu.plant.tut_fi_sql_adapter/Order/in
[FB]Instance Output Topic: activemq:topic:plac/pid_eu.plant.tut_fi_sql_adapter/Order/out
[FB]Instance Direct Output Topic: direct:pid_eu.plant.tut_fi_sql_adapter.Order
```

Figure 27. SQL FB instance initialized console output

4.2.1.1.4 XML Refactor FB Configuration

This FB Type does not require any special entities or attributes to be mentioned while configuration as the XML refactoring logic is encapsulated in the Business logic. This FB type is programmed to extract the data when triggered and performs the refactoring as per the business logic and transmits the refactored XML output. The instance creation of XML Refactor FB Type and its corresponding console output can be witnessed in the pictures below.



Send a JMS Message

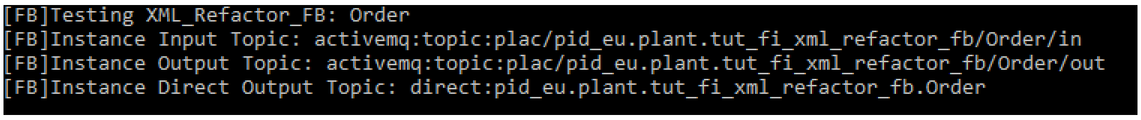
Message Header

Destination	plac/fbm/mgmt/in	Queue or Topic	Topic
Correlation ID		Persistent Delivery	☐
Reply To		Priority	
Type	createFbInstances	Time to live	
Message Group		Message Group Sequence Number	
delay(ms)		Time(ms) to wait before scheduling again	
Number of repeats		Use a CRON string for scheduling	
Number of messages to send	1	Header to store the counter	JMSXMessageCounter

Message body

```
<?xml version="1.0" encoding="UTF-8"?>
<createFbInstances>
   <fbInstance>
      <fbId>Order</fbId>
      <fbPid>pid_eu.plant.tut_fi_xml_refactor_fb</fbPid>
<businessConfiguration>
{"configuration": {
   }
}
</businessConfiguration>
      <inputMessageTypes>
         <inputMessageType>
            <messageTypeName>String</messageTypeName>
            <transformations>
               <transformation>
                  <newMessageTypeName>STRING</newMessageTypeName>
                  <transformationScript>STRING</transformationScript>
               </transformation>
            </transformations>
         </inputMessageType>
      </inputMessageTypes>
      <outputMessageTypes>
         <outputMessageType>
            <messageTypeName>String</messageTypeName>
            <transformations>
               <transformation>
                  <newMessageTypeName>STRING</newMessageTypeName>
                  <transformationScript>STRING</transformationScript>
               </transformation>
            </transformations>
```

Figure 28. XML Refactor FB instance configuration

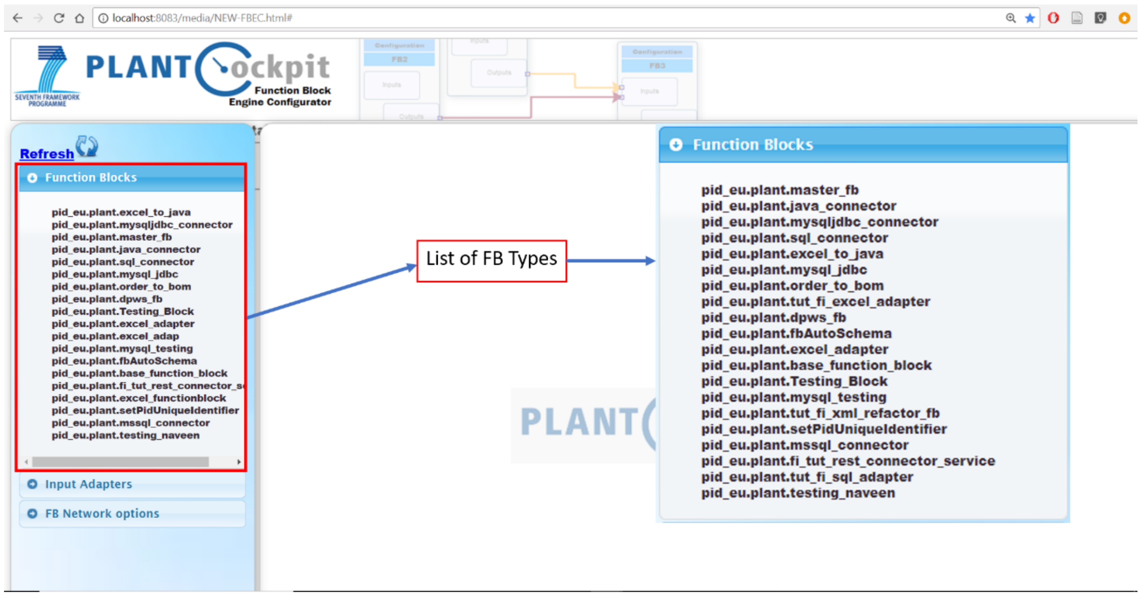


```
[FB]Testing XML_Refactor_FB: Order
[FB]Instance Input Topic: activemq:topic:plac/pid_eu.plant.tut_fi_xml_refactor_fb/Order/in
[FB]Instance Output Topic: activemq:topic:plac/pid_eu.plant.tut_fi_xml_refactor_fb/Order/out
[FB]Instance Direct Output Topic: direct:pid_eu.plant.tut_fi_xml_refactor_fb.Order
```

Figure 29. XML Refactor FB instance initialized console output

4.2.1.2 Instance configuration using FBEC GUI

FBEC comes with a built-in a drag and drop functionality which simplifies the configuration and initiation of an FB instance. The initialized FBs area available on the left pane of the web page. The necessary FBs required to be implemented in the use case is dragged and dropped on the main page. For the implementation of this use case, the URL where the FBEC GUI hosted is http://localhost:8083/media/NEW-FBEC.html#. The home page of FBEC GUI is presented below:



localhost:8083/media/NEW-FBEC.html#

PLANT Cockpit Function Block Engine Configurator

Refresh

Function Blocks

pid_eu.plant.excel_to_java
pid_eu.plant.mysqljdbc_connector
pid_eu.plant.master_fb
pid_eu.plant.java_connector
pid_eu.plant.sql_connector
pid_eu.plant.mysql_jdbc
pid_eu.plant.order_to_bom
pid_eu.plant.dpws_fb
pid_eu.plant.Testing_Block
pid_eu.plant.excel_adapter
pid_eu.plant.excel_adap
pid_eu.plant.mysql_testing
pid_eu.plant.fbAutoSchema
pid_eu.plant.base_function_block
pid_eu.plant.fi_tut_rest_connector_s
pid_eu.plant.excel_functionblock
pid_eu.plant.setPidUniqueIdentifier
pid_eu.plant.mssql_connector
pid_eu.plant.testing_naveen

Input Adapters

FB Network options

List of FB Types

Function Blocks

pid_eu.plant.master_fb
pid_eu.plant.java_connector
pid_eu.plant.mysqljdbc_connector
pid_eu.plant.sql_connector
pid_eu.plant.excel_to_java
pid_eu.plant.mysql_jdbc
pid_eu.plant.order_to_bom
pid_eu.plant.tut_fi_excel_adapter
pid_eu.plant.dpws_fb
pid_eu.plant.fbAutoSchema
pid_eu.plant.excel_adapter
pid_eu.plant.base_function_block
pid_eu.plant.Testing_Block
pid_eu.plant.mysql_testing
pid_eu.plant.tut_fi_xml_refactor_fb
pid_eu.plant.setPidUniqueIdentifier
pid_eu.plant.mssql_connector
pid_eu.plant.fi_tut_rest_connector_service
pid_eu.plant.tut_fi_sql_adapter
pid_eu.plant.testing_naveen

Figure 30. Function Block Engine Configurator

The further steps to create the FB Type instance is straightforward and is similar as to the ActiveMQ based configuration. The vital difference being that FBEC GUI removes the need to copy and paste the Business Configuration message along with business configuration JSON to the ActiveMQ JMS message console. The FBEC GUI gives a dedicated section to provide the business configuration that specific FB Type is designed for and it creates the business configuration message autonomously to provision the FB Type instance. Detailed images of Instance configuration of individual FB via FBEC is presented in Section 4.2.2.

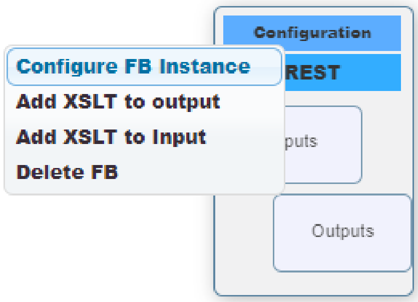


Figure 31. FBEC FB instance creation

4.2.1.3 Routing with Apache Camel

This unified integration solution works hand in hand with Apache Camel which is an integral part of the solution. Apache Camel is responsible for routing the messages between different FB instances, by making a reliable connection from the output node of one FB to the input node of the next in line FB.

This process assists in the interconnection of the FBs and in-turn enables the FBs to communicate with each other. In simple terms, Apache ServiceMix and Apache Camel serve the purpose of /performs Distributed Computing. The below figures represent the connection of routes between different FBs using both ActiveMQ and FBEC GUI.

4.2.1.3.1 Routing with ActiveMQ

The route manager can be accessed with the ActiveMQ topic *"plac/rm/mgmt/in"* as presented in the figure below.

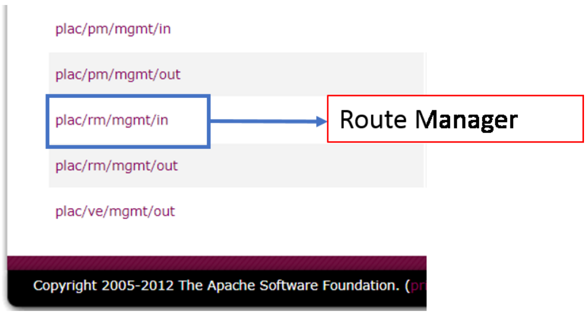
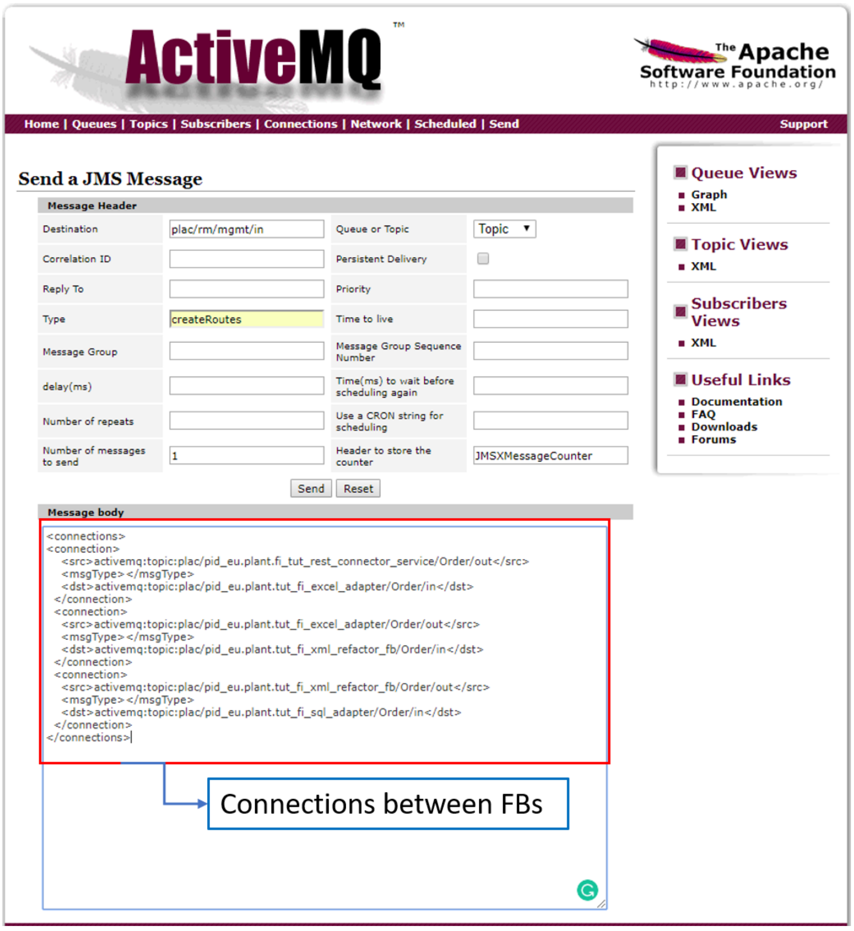

plac/pm/mgmt/in
plac/pm/mgmt/out
plac/rm/mgmt/in
plac/rm/mgmt/out
plac/ve/mgmt/out
Copyright 2005-2012 The Apache Software Foundation. (

Figure 32. ActiveMQ Route Manager

The routes between the different FB instances are defined in the route manager. The routes are described under the *"<connections>" tag*. Each connection between the FBs is specified using the desired source <*src*> and destination <*dst*>. Apache Camel takes care of routing the information during the execution, concerning the routes created. The connections presented in the figure below depicts that, the routes are to be created from the output of the REST FB instance to the input of the Excel Adapter instance. And the output message from the Excel Adapter is routed to the input of XML Refactor FB, and finally, the information of the XML Refactor FB is routed to the SQL Adapter.



Send a JMS Message

Message Header

Destination	plac/rm/mgmt/in	Queue or Topic	Topic
Correlation ID		Persistent Delivery	☐
Reply To		Priority	
Type	createRoutes	Time to live	
Message Group		Message Group Sequence Number	
delay(ms)		Time(ms) to wait before scheduling again	
Number of repeats		Use a CRON string for scheduling	
Number of messages to send	1	Header to store the counter	JMSXMessageCounter

Message body

```
<connections>
<connection>
   <src>activemq:topic:plac/pid_eu.plant.fi_tut_rest_connector_service/Order/out</src>
   <msgType></msgType>
   <dst>activemq:topic:plac/pid_eu.plant.tut_fi_excel_adapter/Order/in</dst>
  </connection>
  <connection>
   <src>activemq:topic:plac/pid_eu.plant.tut_fi_excel_adapter/Order/out</src>
   <msgType></msgType>
   <dst>activemq:topic:plac/pid_eu.plant.tut_fi_xml_refactor_fb/Order/in</dst>
  </connection>
  <connection>
   <src>activemq:topic:plac/pid_eu.plant.tut_fi_xml_refactor_fb/Order/out</src>
   <msgType></msgType>
   <dst>activemq:topic:plac/pid_eu.plant.tut_fi_sql_adapter/Order/in</dst>
  </connection>
</connections>
```

Figure 33. Creating Routes between Function blocks

4.2.1.3.2 Routing with FBEC

The FBEC GUI employs a simple drag and drop feature to create active routes between the function block instances.

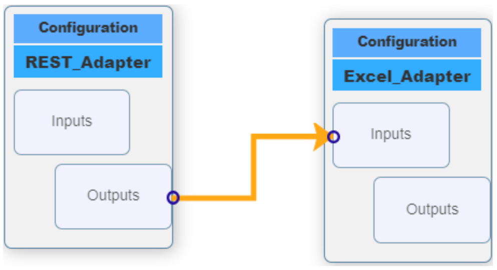


Figure 34. Route creation is FBEC

4.2.2 Use case execution

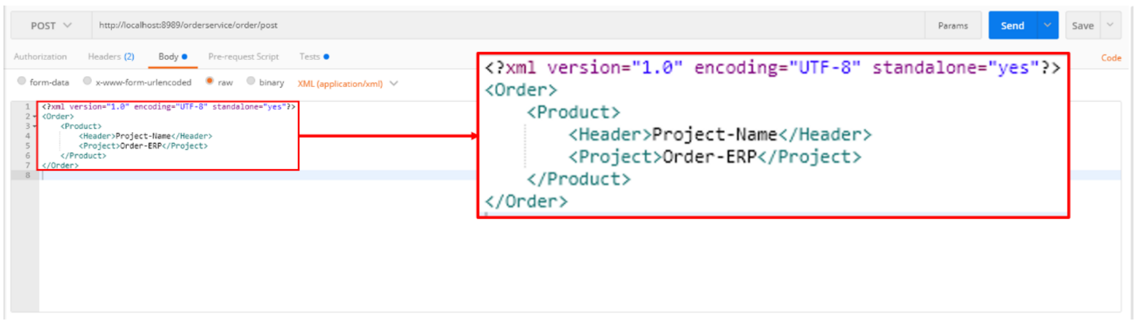
The use case is based on a common scenario in a Manufacturing industry in which the enterprise is required to assign work hours to its employees based on the skill set an also based on the requirement of the project etc. Although the core purpose of this use case and research is to prove the significant integration of PLANTCockpit platform into C2Net architecture, this use case also proves that PLANTCockpit can not only be used for a specialized purpose, but also for every day to day activities across all the level of the organization and provides efficient business process.

For efficient implementation of the use case and seamless integration of heterogeneous systems, an optimized FBN is established and incorporated within PLANTCockpit. By practice, the enterprise allocates work hours to its employee manually as per their criteria and skill set, also considering the already assigned work hours with other existing projects. Using PLANTCockpit, this resource allocation process can be extensively automated to provide efficient results and enhance the business process. All of this can be achieved using the existing legacy systems.

PLANTCockpit enables the enterprise data that are accessible from heterogeneous data sources to be made available across all the levels in the industry. By making the data available, PLANTCockpit facilitates the manufacturing industry to optimize its production planning and enhance its scheduling systems. This use case involves and requires a sequential order of execution of the FBs in the FBN. Each FB are triggered asynchronously one after the other. As soon as the current FB has completed its business logic execution, the output is sent to the next FB in the FBN. The most crucial or the vital factor of this use case execution is the extraction of data from the legacy MS Excel file and especially the database tables and visualizing the data in a format that is acknowledged and proved by the framework, XML in this case. Although the configuration of instance and routes may vary for both methodologies (ActiveMQ & FBEC), the execution remains the same, and both the environments provide similar results, as FBEC is nothing but a configuration tool build to function on top of ActiveMQ and Apache Camel.

4.2.2.1 REST Adapter:

The REST adapter module is deployed along with the other functions blocks in the ESB and is exposed to a specific endpoint and is programmed to listen to a particular port number. Sending a POST request will trigger the first of the series of functions that enables the system to allocate work hours to the employees based on the Availability of the employees and the project timeframe. The REST adapter is configured to accept XML messages, and it has the capability parse specific XML Attribute to determine the content of the POST request. Hence the body of the POST request should be in XML message format.



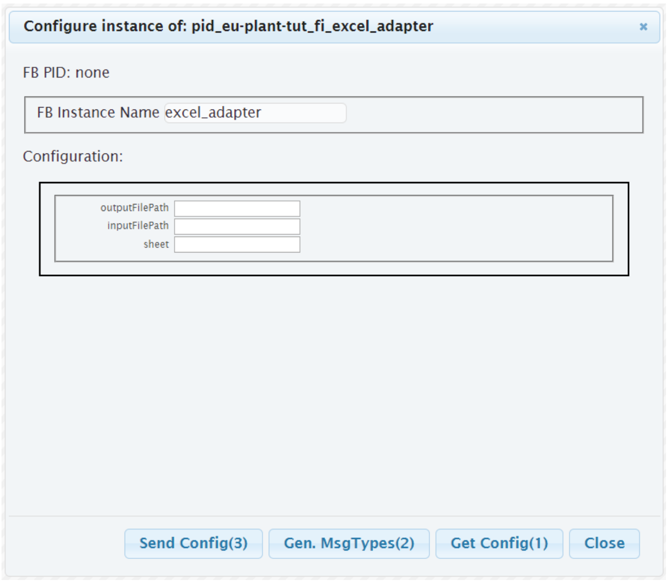
```xml
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<Order>
    <Product>
        <Header>Project-Name</Header>
        <Project>Order-ERP</Project>
    </Product>
</Order>
```

Figure 35. Initiating the Order request

The XML message triggers the REST FB and analyses the message received based on the XML elements and attributes. The REST FB is intelligent to recognize the motivation of the XML and in this case, to create an Order and allocate budgeted work hours to employees. The REST adapter then routes the XML message to the Excel adapter to extract the Project data about the Order received.

4.2.2.2 Excel adapter:

The MS Excel file is the legacy system in which the project data of the industry is stored. Once the Excel FB Type instance is invoked, the excel file is procured based on the input file and path specified during the FB configuration. This file is then parsed into the PLANTCockpit platform. A sample Excel Adapter instance configuration template via FBEC is presented below.



Configure instance of: pid_eu-plant-tut_fi_excel_adapter

FB PID: none

FB Instance Name excel_adapter

Configuration:

outputFilePath
inputFilePath
sheet

Figure 36. FBEC Excel configuration

When the Order is initiated, The FB instance will examine the elements from the Excel FB within the limits of the sheets specified and extracts the project data with respect to the order and transforms the extracted information into a machine-readable XML format that is acknowledged by the next “XML refactor FB” and transmits the data to the same.

By doing this, the Excel FB also saves a copy of the XML data to the destination specified during the FB configuration.

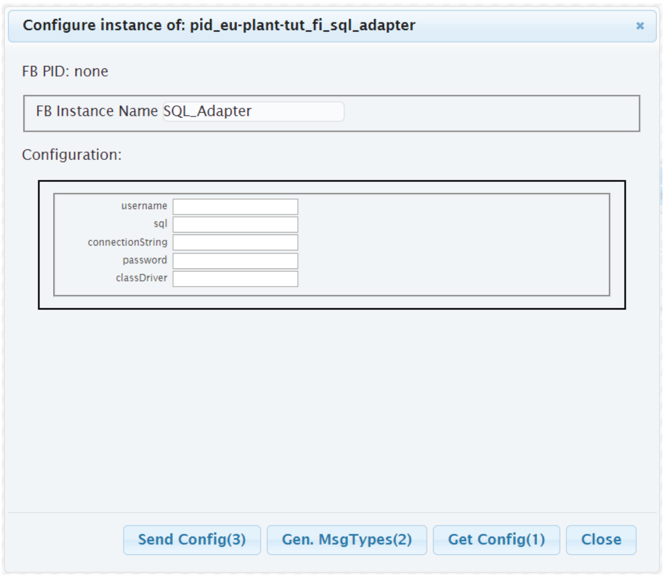
4.2.2.3 XML refactor FB:

In practice, the FB instances deployed in PLANTCockpit gets triggered by the business configuration message. The FB instance is invoked when it receives the business configuration message and that point onward, the FB executes the tasks that it was programmed to function as defined in its business logic. The behavior logic or the functionality of a specific Fb Type can be observed in its “onMessageReceived” method.

The XML refactor FB acts as a gateway between the project data and the Database. The XML FB is an integral part of the system as it validates the data first and then permits the data to be refactored. The structural entities of the project data XML are altered to a more sophisticated format that integrates coherently with the SQL query.

4.2.2.4 SQL Connector:

The SQL connector when triggered initiates a secured connection to a heterogeneous external database (based on the configuration parameters provided during the instance configuration) that resides in an External Legacy system and not within the PLANTCockpit environment. Once the connection is successful, the SQL connector acting as an interaction to the external database system harmonizes the SQL query by incorporating the modified data to the existing query. The consolidated query with the algorithm to allocate work hours is then executed. A sample SQL Adapter instance configuration template via FBEC is presented below.



Configure instance of: pid_eu-plant-tut_fi_sql_adapter

FB PID: none

FB Instance Name SQL_Adapter

Configuration:

username
sql
connectionString
password
classDriver

Figure 37. FBEC SQL Adapter configuration

Once the query is executed, the comprehensive information of available technicians with dedicated skillset is retrieved. The information retrieved can be modularized based on the requirements of the enterprise. The SQL Connector is further modified to interpret the

C2NET internal data structure. The final data retrieved after successful execution of the SQL query is in XML format and can be adequately utilized robustly to manage the enterprise resources. The final output is also saved at the desired destination in a preferred format. The comprehensive function block network employed to implement the use case is represented below.

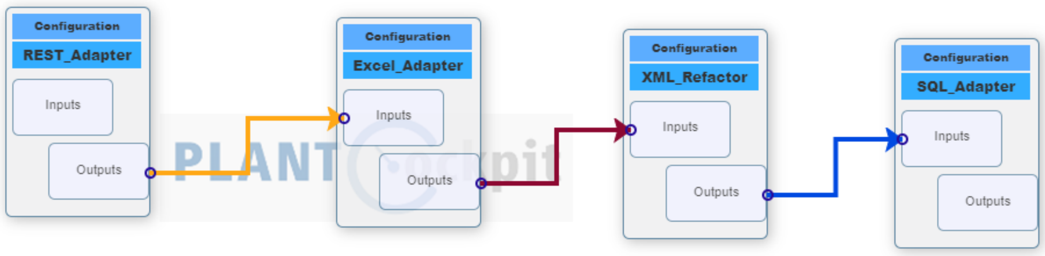


Figure 38. Function Block Network

5. RESULTS

Section 5 supplements to the previous section also discussing the results achieved from the use case execution scenario and provides an overall picture of the output and execution flow of individual function blocks. This section also addresses the problem statements and problem definitions specified in Section 1 and explains how the proposed solution tends to meet the requirements.

5.1 Use case Results

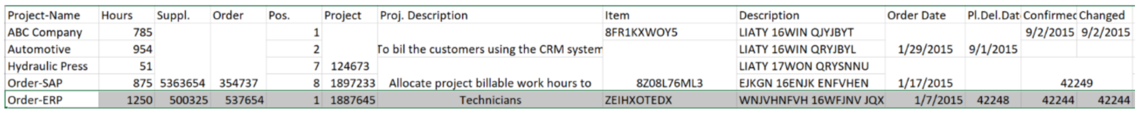
This sub-section demonstrates the execution techniques and the results obtained from individual function blocks which facilitated the information flow also triggering the execution of next in line function blocks.

5.1.1 REST Adapter:

The initialized REST Adapter instance will be listening to the endpoint for requests in *"http://localhost:8989/orderservice/order/post"* Once the request is received, the REST adapter transmits the data, in this case, request is to allocate billable work hours of employees to a project automatically. The REST Adapter accepts XML message body and verifies if the XML element and the attribute correspond to the work hour allocation sequence. The body of the XML data contains the project name of the Order to whom the billable hours are to be allocated. If the XML passes the initial verification, the REST adapter transmits the message to the next function block.

5.1.2 Excel Adapter:

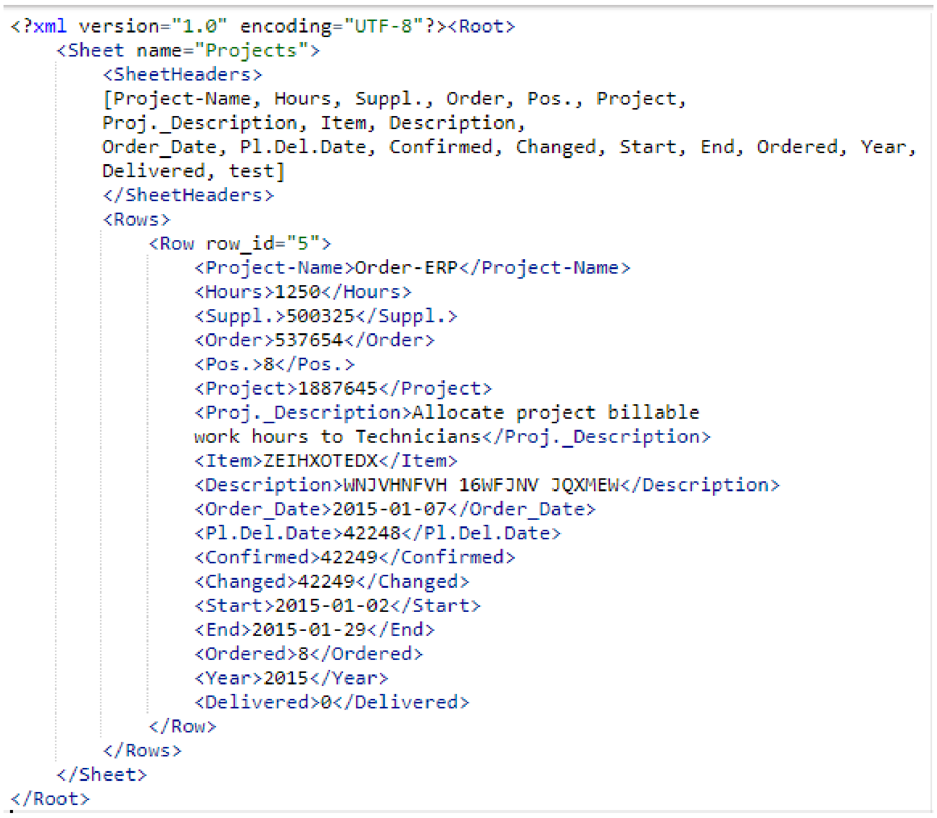
The order received from the REST Adapter is used as an input to trigger the Excel Adapter instance. The Order name received is used as a criterion to query the MS Excel file which is situated in the External legacy system. The excel file to be queried is specified during the Excel FB configuration. The initial step of the excel adapter is to identify the project information from the from the spreadsheet based on the order received. Next, the adapter extracts all the project information, for example, "start date," "end date," "project name" etc. A model excel data used for data extraction is presented below:



Project-Name	Hours	Suppl.	Order	Pos.	Project	Proj. Description	Item	Description	Order Date	Pl.Del.Dat	Confirmec	Changed
ABC Company	785			1			8FR1KXWOY5	LIATY 16WIN QJYJBYT			9/2/2015	9/2/2015
Automotive	954			2		To bil the customers using the CRM system		LIATY 16WIN QRYJBYL	1/29/2015	9/1/2015		
Hydraulic Press	51			7	124673			LIATY 17WON QRYSNNU				
Order-SAP	875	5363654	354737	8	1897233	Allocate project billable work hours to	8Z08L76ML3	EJKGN 16ENJK ENFVHEN	1/17/2015		42249	
Order-ERP	1250	500325	537654	1	1887645	Technicians	ZEIHXOTEDX	WNJVHNFVH 16WFJNV JQX	1/7/2015	42248	42244	42244

Figure 39. Sample Production order Excel spreadsheet

Finally, the critical process executed by the excel adapter is to convert the extracted spreadsheet data to XML format which is represented in the picture below. The excel data information is transported to XML refactor FB. The final output of the Excel Adapter after the data extraction and conversion to XML is represented below:



```
<?xml version="1.0" encoding="UTF-8"?><Root>
    <Sheet name="Projects">
        <SheetHeaders>
        [Project-Name, Hours, Suppl., Order, Pos., Project,
        Proj._Description, Item, Description,
        Order_Date, Pl.Del.Date, Confirmed, Changed, Start, End, Ordered, Year,
        Delivered, test]
        </SheetHeaders>
        <Rows>
            <Row row_id="5">
                <Project-Name>Order-ERP</Project-Name>
                <Hours>1250</Hours>
                <Suppl.>500325</Suppl.>
                <Order>537654</Order>
                <Pos.>8</Pos.>
                <Project>1887645</Project>
                <Proj._Description>Allocate project billable
                work hours to Technicians</Proj._Description>
                <Item>ZEIHXOTEDX</Item>
                <Description>WNJVHNFVH 16WFJNV JQXMEW</Description>
                <Order_Date>2015-01-07</Order_Date>
                <Pl.Del.Date>42248</Pl.Del.Date>
                <Confirmed>42249</Confirmed>
                <Changed>42249</Changed>
                <Start>2015-01-02</Start>
                <End>2015-01-29</End>
                <Ordered>8</Ordered>
                <Year>2015</Year>
                <Delivered>0</Delivered>
            </Row>
        </Rows>
    </Sheet>
</Root>
```

Figure 40. The output of Excel Function Block

5.1.3 XML Refactor FB:

The XML Refactor FB acts as a Gateway between the Excel Adapter and SQL Adapter. The primary characteristics of this FB are to validate the project information XML and filters the applicable information. With the extracted information, the FB finally re-structures the original XML to a more articulated XML format that is relevant and suitable to trigger the SQL Connector.

5.1.4 SQL Connector:

The output of the XML Refactor FB activates the SQL Connector instance. Once the SQL Connector instance is initiated, The JDBC driver incorporated within the SQL Connector tries to establish a secured connection with the external DB legacy system. The DB can be an on-premises DB or the server. The connection is established concerning the parameters specified during the SQL Connector FB Type instance configuration. Once the connection is established, this channel will act as a secure tunnel which facilitates a robust exchange of information between the SQL server and the FB.

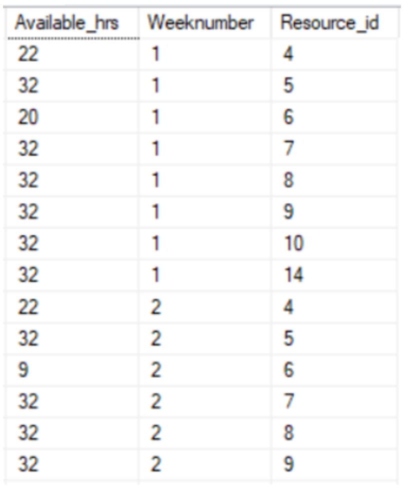
The next course of action is that the trivial data from the re-factored XML structure is cherry-picked and seamlessly incorporated into the pre-defined structured SQL Query. This process makes the SQL query a more comprehensive one and completes the SQL Adapter business logic. The consolidated query is then executed in the legacy system database by the SQL Connector. Once the algorithm incorporated in the SQL query has been executed, available work hours of individual employees is generated.

For demonstration and proving the approach, the query executed is to allocate work hours to technicians of an order for a specific project. The project begins in the starting week of the year 2016 and closes at the fourth week of the same year. Work hours are allocated based on the week numbers. The "start date" and "end date" and the number of "budgeted hours" to be assigned are the most critical data captured by the XML Refactor FB.

The SQL Database Adapter displays the output in 3 fields:

- *ResourceID:* the unique id of the technician
- *WeekNumber:* numerical value of the week number on that year
- *AvailableHours:* the number of hours that specific technician is available for that particular week

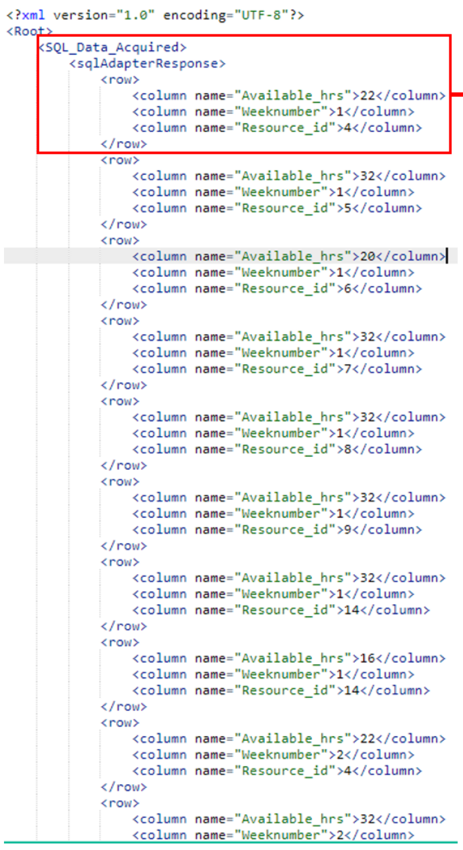
The algorithm queries the database tables which are potential technicians to work on the project with their available hours they can spend on working on this project. The data is queried based on the *WeekNumber* until the end date of the project. The result obtained will be equal and suffice the total work hours of the project. For example, if the project requires 1250 billable hours, the result of the query will possess the data of technicians of a total of 1250 Available hours. The resulting sample database table after a sample query execution is shown below.



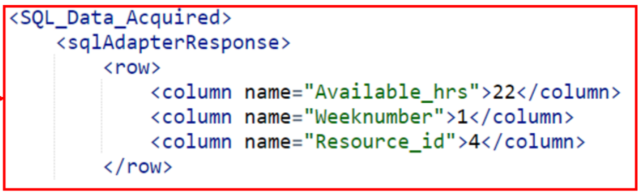
Available_hrs	Weeknumber	Resource_id
22	1	4
32	1	5
20	1	6
32	1	7
32	1	8
32	1	9
32	1	10
32	1	14
22	2	4
32	2	5
9	2	6
32	2	7
32	2	8
32	2	9

Figure 41. Sample Database Table

The subsequent result obtained can be productively utilized by the enterprise to allot the work hours, to the technicians for the respective project order. The data collected from the query is then transmitted back to the SQL Connector, which in-turn converts the information into XML format. The XML data can be saved into any desired location, local or cloud based on the business configuration provided or hardcoded in the FB business logic itself. The XML output obtained from the SQL Adapter is shown below:



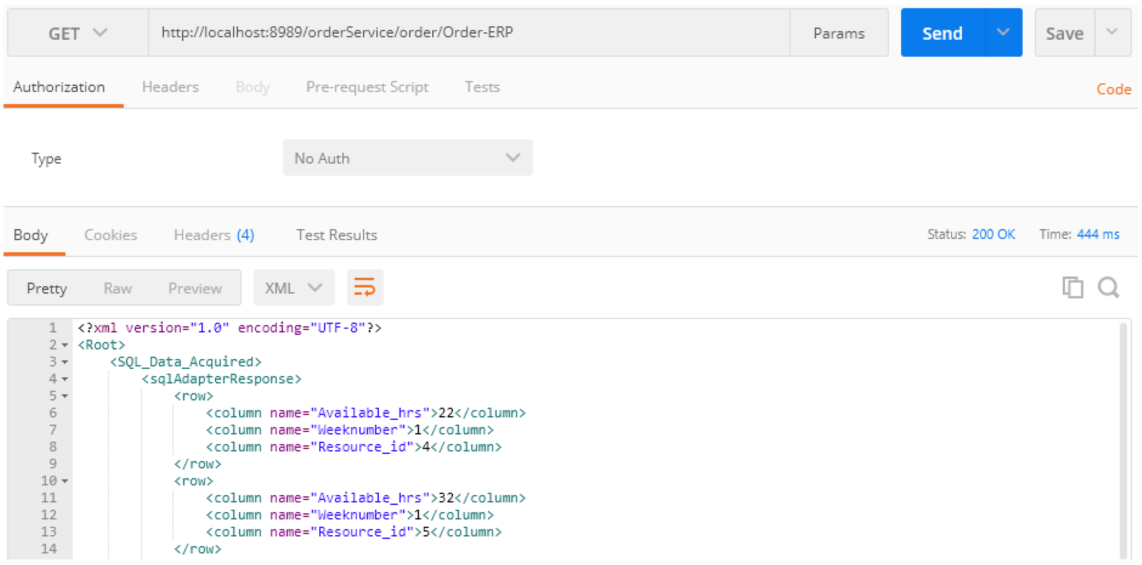
```
<?xml version="1.0" encoding="UTF-8"?>
<Root>
    <SQL_Data_Acquired>
        <sqlAdapterResponse>
            <row>
                <column name="Available_hrs">22</column>
                <column name="Weeknumber">1</column>
                <column name="Resource_id">4</column>
            </row>
            <row>
                <column name="Available_hrs">32</column>
                <column name="Weeknumber">1</column>
                <column name="Resource_id">5</column>
            </row>
            <row>
                <column name="Available_hrs">20</column>
                <column name="Weeknumber">1</column>
                <column name="Resource_id">6</column>
            </row>
            <row>
                <column name="Available_hrs">32</column>
                <column name="Weeknumber">1</column>
                <column name="Resource_id">7</column>
            </row>
            <row>
                <column name="Available_hrs">32</column>
                <column name="Weeknumber">1</column>
                <column name="Resource_id">8</column>
            </row>
            <row>
                <column name="Available_hrs">32</column>
                <column name="Weeknumber">1</column>
                <column name="Resource_id">9</column>
            </row>
            <row>
                <column name="Available_hrs">32</column>
                <column name="Weeknumber">1</column>
                <column name="Resource_id">14</column>
            </row>
            <row>
                <column name="Available_hrs">16</column>
                <column name="Weeknumber">1</column>
                <column name="Resource_id">14</column>
            </row>
            <row>
                <column name="Available_hrs">22</column>
                <column name="Weeknumber">2</column>
                <column name="Resource_id">4</column>
            </row>
            <row>
                <column name="Available_hrs">32</column>
                <column name="Weeknumber">2</column>
```



```
<SQL_Data_Acquired>
    <sqlAdapterResponse>
        <row>
            <column name="Available_hrs">22</column>
            <column name="Weeknumber">1</column>
            <column name="Resource_id">4</column>
        </row>
```

Figure 42. Output of SQL Adapter query

The SQL Adapter query outputs the XML which can be utilized efficiently to allocate work hours to employees. The resulting query contains the "*ResourceID*," "*WeekNumber*" and "*AvailableHours*." The final output is saved on the local server and thanks to Apache CXF, the output can be retrieved by querying ("GET" request) the respective endpoint as shown in the below picture. The Apache CXF identifies the data to be fetched based on the name of the Order implemented, in this case *"Order-ERP"*.



http://localhost:8989/orderService/order/Order-ERP

Type: No Auth

Status: 200 OK Time: 444 ms

```
<?xml version="1.0" encoding="UTF-8"?>
<Root>
    <SQL_Data_Acquired>
        <sqlAdapterResponse>
            <row>
                <column name="Available_hrs">22</column>
                <column name="Weeknumber">1</column>
                <column name="Resource_id">4</column>
            </row>
            <row>
                <column name="Available_hrs">32</column>
                <column name="Weeknumber">1</column>
                <column name="Resource_id">5</column>
            </row>
```

Figure 43. Querying the final SQL output using REST client

5.2 Learnings & Outcome

This research addresses the problem statements posed in Section 1.2.1 by exhibiting an approach to integrate legacy by adopting FB based techniques. The FB-based methodology encourages the robust enterprise system integration and also the heterogeneity of the framework is addressed. Data logistics is also a vital characteristic to consider when it comes to achieving seamless enterprise integration across all the levels of Automation pyramid. All the requirements above are fulfilled by incorporating the PLANTCockpit OS in the approach. The consolidated reference architecture of the use case implementation is presented in the diagram below:

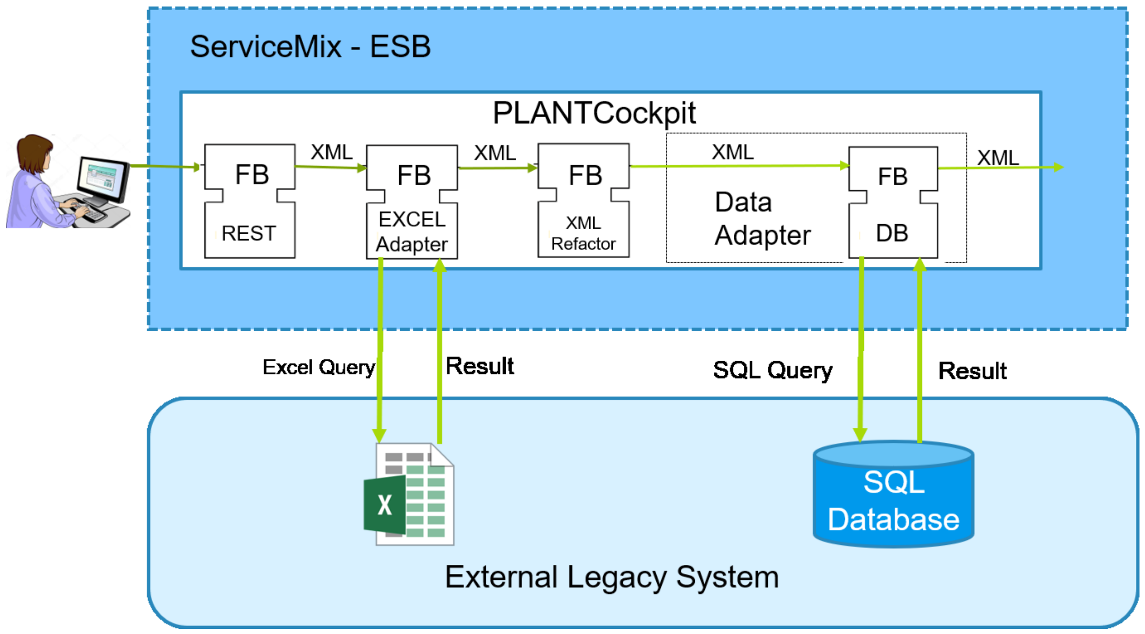


Figure 44. USE CASE reference architecture diagram

PLANTCockpit OS is an efficient platform to facilitate dynamic and real-time data collection, data transformation and data logistics form heterogeneous data sources. Consequently, this use case approach proposed in this research presents the implementation of PLANTCockpit OS which can be used as a proof of concept to be employed in C2NET reference architecture as a conceivable solution to empower a flexible and adaptable data collection solution.

6. CONCLUSION & FUTURE RESEARCH

This concluding chapter of the thesis tends to look back on the solution presented and provides a summary of the scope of the thesis. Probable future research extension is also discussed. Also, this concluding section provides personal insights and thoughts on this research work.

This thesis addresses the problem statements described in Section 1.2.1 and provides a comprehensive solution which solves not only the problem statement presented in this thesis but also this solution offers a methodology which can facilitate seamless enterprise integration which is much required in the manufacturing enterprises these days.

The FB Adapters and Transformation components developed and deployed in the PLANTCockpit framework establish a potential connection with heterogeneous external legacy systems. The objective here was to bring together the data from these components using PLANTCockpit as an interface. PLANTCockpit serves as a scintillating platform to work with heterogeneous data sources, revamping data-logistics by transforming data of disparate data types into something platform-specific to be processed further by Function Blocks.

Applying the IEC61499 reference architecture efficiently, the solution is successfully designed and implemented. The plug-and-play loosely coupled entities are characterized to be flexible, reusable and easily configurable during run-time. The pursued approach even though presents only a baseline framework of the functional traits of the Function block, the Function block, in general, possesses discrete exotic features which enrich the functionalities range of the IEC61499 Function block standard. Adopting Function block structure has contributed to numerous improvements to the framework such as providing extensibility, improving interoperability, reducing the complexity of the Application and enhancing the process orchestration of the system.

During this research, a comprehensive solution to enable Enterprise system integration with robust data collection and data-logistics has been presented by incorporating an innovative platform called PLANTCockpit. By adapting to and implementing this solution, enterprises can empower an integrated and a comprehensive view of the industrial processes such as manufacturing and logistics. PLANTCockpit proposes the development and implementation of a layered architecture solution which facilitates harmonious and consistent access to the factory information systems.

The prototype execution of the use case has demonstrated extraordinary features and functionalities of this framework makes it feasible to implement in diverse scenarios and enterprises, thus allowing flexible analysis of shop floor data and visualization of information flow. PLANTCockpit solution enables a prominent level of transparency for the business process and permits a new level of optimization in manufacturing process amidst heterogeneous information systems.

6.1 Future Work

Function blocks are designed to be flexible, extensible and re-usable. On owing to their plug-and-play mechanisms, these modular function blocks can be integrated and extended to heterogeneous external data source systems.

Although the scope of this thesis research is restricted to the functionality and implementation of four function blocks, there is always room for improvisation and improvement. One such example could be an expansion of the use case presented with a Device Profile for Web Services (DPWS) function block. The function block would work on the publish-subscribe mechanism and can listen to shop floor devices and can subscribe to its events and publish information, everything based on its FB Type and its configuration. The implementation of DPWS adapter to fetch shop-floor data was conceived in the later stages of the thesis but was discarded due to time constraints.

Another critical aspect which can be taken into consideration is the security of the consolidated framework. As of now, the security of the solution depends on the enterprise implementing the framework. But, the solution would be more reliable if there exists an extra layer of protection over the existing one to enable secured communication between the systems. For example, SSH tunneling can be implemented to enhance the client-server interface and adding an extra layer of encryption to the legacy systems. Web sockets and authentication token technologies can be applied to strengthen the exchange of information between the enterprise systems.

Another common factor which can be improved is streamlining the verbosity of the business logic of the FB Adapters to increase their flexibility by upgrading the business configuration functionalities and enhance their interoperability with heterogeneous systems. The consolidated Function block networks can be built more versatile to adapt the rapid changes of manufacturing enterprise environment in real-time. These characteristics and functionalities can be achieved by implementing comprehensive FB adapters and robust function blocks with complex business logic.

REFERENCES

[1] "The Industrial Revolution," *The British Library*. [Online]. Available: https://www.bl.uk/georgian-britain/articles/the-industrial-revolution. [Accessed: 10-Feb-2018].

[2] "Manufacturing Industry," *EconomyWatch*. [Online]. Available: http://www.economywatch.com/world-industries/manufacturing. [Accessed: 10-Feb-2018].

[3] "Manufacturing statistics - NACE Rev. 2 - Statistics Explained." [Online]. Available: http://ec.europa.eu/eurostat/statistics-explained/index.php/Manufacturing_statistics_-_NACE_Rev._2. [Accessed: 10-Feb-2018].

[4] "File:Sectoral fig1 analysis of Manufacturing (NACE Section C), EU-28, 2014 .png - Statistics Explained." [Online]. Available: http://ec.europa.eu/eurostat/statistics-explained/index.php/File:Sectoral_fig1_analysis_of_Manufacturing_(NACE_Section_C),_EU-28,_2014_.png. [Accessed: 10-Feb-2018].

[5] "Official Documents - Horizon 2020 - European Commission," *Horizon 2020*. [Online]. Available: /programmes/horizon2020/en/official-documents. [Accessed: 11-Feb-2018].

[6] "Industrial production statistics - Statistics Explained." [Online]. Available: http://ec.europa.eu/eurostat/statistics-explained/index.php/Industrial_production_statistics. [Accessed: 11-Feb-2018].

[7] F. Zezulka, P. Marcon, I. Vesely, and O. Sajdl, "Industry 4.0 – An Introduction in the phenomenon," *IFAC-Pap.*, vol. 49, no. 25, pp. 8–12, Jan. 2016.

[8] H. S. Kang *et al.*, "Smart manufacturing: Past research, present findings, and future directions," *Int. J. Precis. Eng. Manuf.-Green Technol.*, vol. 3, no. 1, pp. 111–128, Jan. 2016.

[9] Y. Lu, "Industry 4.0: A survey on technologies, applications and open research issues," *J. Ind. Inf. Integr.*, vol. 6, pp. 1–10, Jun. 2017.

[10] S. Ghimire, R. Melo, J. Ferreira, C. Agostinho, and R. Goncalves, "Continuous Data Collection Framework for Manufacturing Industries," in *On the Move to Meaningful Internet Systems: OTM 2015 Workshops*, vol. 9416, I. Ciuciu, H. Panetto, C. Debruyne, A. Aubry, P. Bollen, R. Valencia-García, A. Mishra, A. Fensel, and F. Ferri, Eds. Cham: Springer International Publishing, 2015, pp. 29–40.

[11] N. Govindarajan, B. R. Ferrer, X. Xu, A. Nieto, and J. L. M. Lastra, "An approach for integrating legacy systems in the manufacturing industry," in *2016 IEEE 14th International Conference on Industrial Informatics (INDIN)*, 2016, pp. 683–688.

[12] S. Cavalieri, F. D. Urso, C. Floridia, and A. Rossettini, "Definition of a middleware for the vertical and horizontal integration of information in large plants," in *Industrial Technology, 2004. IEEE ICIT'04. 2004 IEEE International Conference on*, 2004, vol. 2, pp. 745–750.

[13] "Innovation in Manufacturing - Research & Innovation - Key Enabling Technologies - European Commission." [Online]. Available: http://ec.europa.eu/research/industrial_technologies/innovation-in-manufacturing_en.html. [Accessed: 04-Feb-2018].

[14] *A 'Manufacturing Imperative' in the EU – Europe's Position in Global Manufacturing and the Role of Industrial Policy (publication)*. .

[15] C. Brecher *et al.*, “Integrative Production Technology for High-wage Countries,” in *Integrative Production Technology for High-Wage Countries*, Springer, Berlin, Heidelberg, 2012, pp. 17–76.
[16] “Demystifying real-time manufacturing data acquisition solutions,” *LinkedIn Pulse*, 18-Sep-2014. [Online]. Available: https://www.linkedin.com/pulse/20140918054455-5235307-demystifying-real-time-manufacturing-data-acquisition-solutions. [Accessed: 18-Feb-2016].
[17] “Turn machine data into insights to see into your business & IT systems,” *Splunk*. [Online]. Available: http://www.splunk.com/en_us/resources/operational-intelligence.html. [Accessed: 19-Feb-2016].
[18] J. Kletti, Ed., *Manufacturing Execution Systems — MES*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2007.
[19] A. Bratukhin and T. Sauter, “Functional Analysis of Manufacturing Execution System Distribution,” *IEEE Trans. Ind. Inform.*, vol. 7, no. 4, pp. 740–749, Nov. 2011.
[20] H. Wang, L. Liu, Y. Fei, and T. Liu, “A Collaborative Manufacturing Execution System Oriented to Discrete Manufacturing Enterprises,” in *Cooperative Design, Visualization, and Engineering*, 2014, pp. 277–285.
[21] “MESA International,” 19-Oct-2017. [Online]. Available: http://www.mesa.org/en/index.asp. [Accessed: 02-Apr-2018].
[22] “MESA - Manufacturing Enterprise Solutions Association | ATS.” [Online]. Available: http://www.ats-global.com/mesa-manufacturing-enterprise-solutions-association_95_gben. [Accessed: 02-Apr-2018].
[23] D. Brandl and P. Owen, “Manufacturing Operations Management,” *Univ. Camb. Inst. Manuf.*, 2003.
[24] V. Filipov and N. Christova, “A Solution for Integrated Manufacturing Operation Management,” in *2008 International Conference on Computational Intelligence for Modelling Control Automation*, 2008, pp. 527–532.
[25] “What is Manufacturing Operations Management?: Siemens PLM Software.” [Online]. Available: https://www.plm.automation.siemens.com/mom/what-is-mom/manufacturing-operations-management.shtml. [Accessed: 20-Feb-2016].
[26] “Swiftcourse Technologies - MES - Manufacturing Execution System.” [Online]. Available: http://swiftcourse.com/TodaysManufacturing.html. [Accessed: 02-Apr-2018].
[27] Q. Han, S. Mehrotra, and N. Venkatasubramanian, “Energy efficient data collection in distributed sensor environments,” in *24th International Conference on Distributed Computing Systems, 2004. Proceedings*, 2004, pp. 590–597.
[28] A. Dedinak, G. Kronreif, and C. Wogerer, “Vertical integration of production systems,” in *Industrial Technology, 2003 IEEE International Conference on*, 2003, vol. 1, pp. 376–380.
[29] J. M. M. Rumbold and B. K. Pierscionek, “What Are Data? A Categorization of the Data Sensitivity Spectrum,” *Big Data Res.*, Dec. 2017.
[30] “Definition of DATA.” [Online]. Available: https://www.merriam-webster.com/dictionary/data. [Accessed: 02-Apr-2018].
[31] Losee Robert M., “A discipline independent definition of information,” *J. Am. Soc. Inf. Sci.*, vol. 48, no. 3, pp. 254–269, Dec. 1998.
[32] M. Leite and V. Braz, “Agile manufacturing practices for new product development: industrial case studies,” *J. Manuf. Technol. Manag. Bradf.*, vol. 27, no. 4, pp. 560–576, 2016.
[33] J. Goodwin and M. Woodfield, “2006 IPCC Guidelines for National Greenhouse Gas Inventories- APPROACHES TO DATA COLLECTION.” .

[34] Y. Li, J. Rong, and S. Huang, “Technology study of new industry data collection and monitoring system,” in *Electronic and Mechanical Engineering and Information Technology (EMEIT), 2011 International Conference on*, 2011, vol. 9, pp. 4709–4712.

[35] H. H. Jian, C. R. Xiong, and G. B. Huang, “Design of a Data Collection System in Manufacturing Informationization,” *Appl. Mech. Mater. Zurich*, vol. 236–237, p. 283, Nov. 2012.

[36] D. J. B. Aguilar and others, “Manufacturing equipment data collection framework,” 2012.

[37] “Production data acquisition systems with Plant iT and brewmaxx.” [Online]. Available: http://www.proleit.com/solutions/production-data-acquisition-pda.html. [Accessed: 02-Dec-2015].

[38] N. US Department of Commerce, “Real-Time Data Analytics for Smart Manufacturing Systems Project.” [Online]. Available: http://www.nist.gov/el/msid/lifecycle/rtdasms.cfm. [Accessed: 01-Dec-2015].

[39] A. Kadadi, R. Agrawal, C. Nyamful, and R. Atiq, “Challenges of data integration and interoperability in big data,” in *Big Data (Big Data), 2014 IEEE International Conference on*, 2014, pp. 38–40.

[40] L. Zhang, “Integration and collection of heterogeneous data based on metedata,” in *Information Management, Innovation Management and Industrial Engineering (ICIII), 2013 6th International Conference on*, 2013, vol. 1, pp. 205–208.

[41] F. Martincic and L. Schwiebert, “Adaptive data collection in sensor networks,” in *Wireless Days (WD), 2009 2nd IFIP*, 2009, pp. 1–6.

[42] “Understanding Enterprise Application Integration - The Benefits of ESB for EAI,” *MuleSoft*, 16-Dec-2014. [Online]. Available: https://www.mulesoft.com/resources/esb/enterprise-application-integration-eai-and-esb. [Accessed: 11-Mar-2018].

[43] F. B. Vernadat, “Enterprise Modelling and Integration,” in *Enterprise Inter- and Intra-Organizational Integration*, Springer, Boston, MA, 2003, pp. 25–33.

[44] D. R. Ferreira, *Enterprise Systems Integration: A Process-Oriented Approach*. Springer Science & Business Media, 2013.

[45] A. Hänninen, “Enterprise Integration Patterns in Service Oriented Systems,” Jun. 2014.

[46] D. Chen, G. Doumeingts, and F. Vernadat, “Architectures for enterprise integration and interoperability: Past, present and future,” *Comput. Ind.*, vol. 59, no. 7, pp. 647–659, Sep. 2008.

[47] O. Schaefer and C. Hsu, “Scalable integration: from enterprise information management to real time process control using the metadatabase model,” in *Proceedings of IEEE International Conference on Systems, Man and Cybernetics*, 1994, vol. 2, pp. 1860–1867 vol.2.

[48] I. Harjunkoski, R. Nyström, and A. Horch, “Integration of scheduling and control—Theory or practice?,” *Comput. Chem. Eng.*, vol. 33, no. 12, pp. 1909–1918, Dec. 2009.

[49] “Automation Pyramid.” [Online]. Available: https://www.smctraining.com/en/webpage/indexpage/312. [Accessed: 10-Mar-2018].

[50] “Joined PLCopen and OPC Foundation technical working group releases next step in transparent communication,” *Automation Inside*. .

[51] “Technology ISA-95,” *ISA 95*. [Online]. Available: https://isa-95.com/technology-isa95/. [Accessed: 11-Mar-2018].

[52] P. Ashmore, “1SA95 Standards - Practical Application Manufacturing,” in *2006 IET Seminar on Enterprise Integration of Control Systems*, 2006, pp. 81–96.

[53] “ISA-95.01 Models & Terminology,” *ISA 95*. [Online]. Available: https://isa-95.com/isa-95-01-models-terminology/. [Accessed: 17-Mar-2018].

[54] “ISA-95.02 Object Model Attributes,” *ISA 95*. [Online]. Available: https://isa-95.com/isa-95-02-object-model-attributes/. [Accessed: 17-Mar-2018].

[55] “ISA-95.03 Activity Models,” *ISA 95*. [Online]. Available: https://isa-95.com/isa-95-03-activity-models/. [Accessed: 17-Mar-2018].

[56] “ISA-95.04 Object Models & Attributes,” *ISA 95*. [Online]. Available: https://isa-95.com/isa-95-04-object-models-attributes/. [Accessed: 17-Mar-2018].

[57] “ISA-95.05 B2M Transactions,” *ISA 95*. [Online]. Available: https://isa-95.com/isa-95-05-b2m-transactions/. [Accessed: 17-Mar-2018].

[58] C. Verdouw, R. Robbemond, and J. W. Kruize, “Integration of Production Control and Enterprise Management Systems in Horticulture,” 2015.

[59] O. Givehchi, K. Landsdorf, P. Simoens, and A. W. Colombo, “Interoperability for Industrial Cyber-Physical Systems: An Approach for Legacy Systems,” *IEEE Trans. Ind. Inform.*, vol. 13, no. 6, pp. 3370–3378, Dec. 2017.

[60] “ISA-95, Enterprise-Control System Integration,” *ISA 95*. [Online]. Available: http://isa-95.com/. [Accessed: 05-Feb-2016].

[61] T. J. Williams, “The Purdue enterprise reference architecture,” *Comput. Ind.*, vol. 24, no. 2, pp. 141–158, Sep. 1994.

[62] ISA (Society), *Enterprise-control system integration. models and terminology.* Research Triangle Park, N.C.: ISA, 2000.

[63] D. M. Neumann, “Evolution process for legacy system transformation,” in *IEEE Technical Applications Conference. Northcon/96. Conference Record*, 1996, pp. 57–62.

[64] A. (Nick) Dedeke, “Improving Legacy-System Sustainability: A Systematic Approach,” *IT Prof. Mag. Wash.*, vol. 14, no. 1, pp. 38–43, Feb. 2012.

[65] “What is a Legacy System (in Computing)? - Definition from Techopedia,” *Techopedia.com*. [Online]. Available: https://www.techopedia.com/definition/635/legacy-system. [Accessed: 18-Mar-2018].

[66] Nelson H. Weiderman, John K. Bergey, Dennis B. Smith, and Scott R. Tilley, “Approaches to Legacy System Evolution,” Dec. 1997.

[67] S. Jha, M. Jha, L. O’Brien, and M. Wells, “Integrating legacy system into big data solutions: Time to make the change,” in *2014 Asia-Pacific World Congress on Computer Science and Engineering (APWC on CSE)*, 2014, pp. 1–10.

[68] W. Qifeng, “Semantic Framework Model-Based Intelligent Information System Integration Mode for Manufacturing Enterprises,” in *Second International Symposium on Intelligent Information Technology Application, 2008. IITA ’08*, 2008, vol. 1, pp. 223–227.

[69] M. van Steen and A. S. Tanenbaum, “A brief introduction to distributed systems,” *Computing*, vol. 98, no. 10, pp. 967–1009, Oct. 2016.

[70] “Overlay Networks: Toward Information Networking.,” *CRC Press*, 09-Feb-2010. [Online]. Available: https://www.crcpress.com/Overlay-Networks-Toward-Information-Networking/Tarkoma/p/book/9781439813713. [Accessed: 24-Mar-2018].

[71] O. Kašpar and M. Radakovič, “Distributed industrial automation systems: Automated communication patterns recognition,” in *2011 IEEE International Conference on Industrial Technology*, 2011, pp. 262–269.

[72] P. Helo, M. Suorsa, Y. Hao, and P. Anussornnitisarn, “Toward a cloud-based manufacturing execution system for distributed manufacturing,” *Comput. Ind.*, vol. 65, no. 4, pp. 646–656, May 2014.

[73] S. K. Makki, “The Integration and Interoperability Issues of Legacy and Distributed Systems,” in *Seventh International Conference on Web-Age Information Management Workshops, 2006. WAIM ’06*, 2006, pp. 21–21.

[74] R. Perrey and M. Lycett, “Service-oriented architecture,” in *2003 Symposium on Applications and the Internet Workshops, 2003. Proceedings.*, 2003, pp. 116–119.

[75] N. Gold, A. Mohan, C. Knight, and M. Munro, “Understanding service-oriented software,” *IEEE Softw.*, vol. 21, no. 2, pp. 71–77, Mar. 2004.

[76] A. Girbea, C. Suciu, S. Nechifor, and F. Sisak, “Design and Implementation of a Service-Oriented Architecture for the Optimization of Industrial Applications,” *IEEE Trans. Ind. Inform.*, vol. 10, no. 1, pp. 185–196, Feb. 2014.

[77] Z. Laliwala and S. Chaudhary, “Event-driven Service-Oriented Architecture,” in *2008 International Conference on Service Systems and Service Management*, 2008, pp. 1–6.

[78] “Combining Service-Oriented Architecture and Event-Driven Architecture using an Enterprise Service Bus,” 28-Mar-2006. [Online]. Available: http://www.ibm.com/developerworks/library/ws-soa-eda-esb/index.html. [Accessed: 25-Mar-2018].

[79] N. Serrano, J. Hernantes, and G. Gallardo, “Service-Oriented Architecture and Legacy Systems,” *IEEE Softw.*, vol. 31, no. 5, pp. 15–19, Sep. 2014.

[80] J. McGovern, Ed., *Enterprise service oriented architectures: concepts, challenges, recommendations*. Dordrecht, The Netherlands: Springer, 2006.

[81] J. Christensen, T. Strasser, A. Valentini, V. Vyatkin, and A. Zoitl, *The IEC 61499 Function Block Standard: Overview of the Second Edition*. 2012.

[82] R. W. Lewis and I. of E. Engineers, *Modelling control systems using IEC 61499: applying function blocks to distributed systems*. Institution of Electrical Engineers, 2001.

[83] C. Sunder *et al.*, “Usability and Interoperability of IEC 61499 based distributed automation systems,” in *2006 4th IEEE International Conference on Industrial Informatics*, 2006, pp. 31–37.

[84] K.-H. John and M. Tiegelkamp, *IEC 61131–3: Programming Industrial Automation Systems: Concepts and Programming Languages, Requirements for Programming Systems, Aids to Decision-Making Tools*. Berlin Heidelberg: Springer-Verlag, 2001.

[85] T. Strasser *et al.*, “Framework for Distributed Industrial Automation and Control (4DIAC),” in *2008 6th IEEE International Conference on Industrial Informatics*, 2008, pp. 283–288.

[86] T. Strasser, A. Zoitl, J. Christensen, and C. Sünder, “Design and Execution Issues in IEC 61499 Distributed Automation and Control Systems,” *IEEE Trans. Syst. Man Cybern. Part C Appl. Rev.*, vol. 41, pp. 41–51, Feb. 2011.

[87] W. Dai and V. Vyatkin, “Redesign Distributed PLC Control Systems Using IEC 61499 Function Blocks,” *IEEE Trans. Autom. Sci. Eng.*, vol. 9, no. 2, pp. 390–401, Apr. 2012.

[88] “PLANTCockpit White-Paper.” [Online]. Available: https://cordis.europa.eu/docs/projects/cnect/8/260018/080/deliverables/001-PLANTCockpitD33V10.pdf.

[89] “PLANTCockpit Open Source: Solution.” [Online]. Available: http://www.tut.fi/plantcockpit-os/Solution.html. [Accessed: 22-Dec-2015].

[90] V. Vasyutynskyy, C. Hengstler, J. McCarthy, K. G. Brennan, D. Nadoveza, and A. Dennert, “Layered architecture for production and logistics cockpits,” in *2012 IEEE 17th Conference on Emerging Technologies Factory Automation (ETFA)*, 2012, pp. 1–9.

[91] A. Dennert, J. G. I. Montemayor, J. Krause, S. Hesse, J. L. M. Lastra, and M. Wollschlaeger, “Advanced Concepts for Flexible Data Integration in Heterogeneous Production Environments,” *IFAC Proc. Vol.*, vol. 46, no. 7, pp. 348–353, May 2013.

[92] A. Dennert, A. Gössling, J. Krause, M. Wollschlaeger, and A. M. H. Montoya, “Vertical data integration in automation based on IEC 61499,” in *Factory Communication Systems (WFCS), 2012 9th IEEE International Workshop on*, 2012, pp. 99–102.

[93] S. Iarovyi, J. Garcia, and J. L. M. Lastra, “An approach for OSGi and DPWS interoperability: Bridging enterprise application with shop-floor,” in *2013 11th IEEE International Conference on Industrial Informatics (INDIN)*, 2013, pp. 200–205.

[94] “D3.3 - PLANTCockpit White Paper - PLANTCockpit_D3.3_Public.pdf.” [Online]. Available: http://www.plantcockpit.eu/fileadmin/PLANTCOCKPIT/user_upload/PLANTCockpit_D3.3_Public.pdf. [Accessed: 16-Dec-2015].

[95] N. Sheble, “IEC 62264 and ISA-95: Enterprise-control system integration,” *InTech Durh.*, vol. 54, no. 3, p. 62, Mar. 2007.

[96] P. Ziegler and K. R. Dittrich, “Data Integration — Problems, Approaches, and Perspectives,” in *Conceptual Modelling in Information Systems Engineering*, J. Krogstie, A. L. Opdahl, and S. Brinkkemper, Eds. Springer Berlin Heidelberg, 2007, pp. 39–58.

[97] “Overcoming Data Integration Problems,” *InfoManagement Direct*. [Online]. Available: http://www.information-management.com/infodirect/19980701/overcoming-data-integration-problems-924-1.html. [Accessed: 02-Dec-2015].

[98] D. Cunningham, “Cloud Data Integration Best Practices.” [Online]. Available: http://www.slideshare.net/dcunni07/cloud-data-integration-best-practices. [Accessed: 02-Dec-2015].

[99] M. Syafrudin, N. L. Fitriyani, D. Li, G. Alfian, J. Rhee, and Y.-S. Kang, “An Open Source-Based Real-Time Data Processing Architecture Framework for Manufacturing Sustainability,” *Sustainability*, vol. 9, no. 11, p. 2139, Nov. 2017.

[100] “Business Intelligence in Manufacturing Sector | Helical Tech.” .

[101] R. T. Qu, R. Lim, Z. Q. Ding, H. He, A. Aendenroomer, and K. M. Goh, “Distributed control application platform-a control platform for advanced manufacturing systems,” in *Proceedings 2003 IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM 2003)*, 2003, vol. 2, pp. 1166–1171 vol.2.

[102] V. Vyatkin, *IEC 61499 Function Blocks for Embedded and Distributed Control Systems Design*. 2006.

[103] “Welcome to Apache ServiceMix!” [Online]. Available: http://servicemix.apache.org/index.html. [Accessed: 08-Apr-2018].

[104] G. Hohpe and B. Woolf, *Enterprise Integration Patterns: Designing, Building, and Deploying Messaging Solutions*. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc., 2003.

[105] “Apache POI - the Java API for Microsoft Documents.” [Online]. Available: https://poi.apache.org/. [Accessed: 31-Mar-2018].