A Multi-agent Approach for Processing Industrial Enterprise Data

Wael M. Mohammed, Borja Ramis Ferrer, Jose L. Martinez Lastra
Tampere University of Technology
Tampere, Finland
{wael.mohammed, borja.ramisferrer, jose.lastra}@tut.fi

Raquel Sanchis, Beatriz Andres
Research Centre on Production Management and Engineering (CIGIP)
Universitat Politècnica de València
València, Spain
{rsanchis, bandres}@cigip.upv.es

Carlos Agostinho
Centre of Technology and Systems, CTS, UNINOVA
Caparica, Portugal
ca@uninova.pt

***Abstract*— The C2NET project aims to provide cloud-based platform for the supply chain interactions. The architecture of such platform includes a Data Collection Framework (DCF) for managing the collection of the company's data. The DCF collects, transforms and stores data from both Internet of Things (IoT) devices in the factory shopfloor and company enterprises data via two types of hub; Legacy system hub (LSH) and IoT hub. Since the C2NET, targets the Small and Medium-sized Enterprises (SMEs), the enterprise data, or legacy data as called in the C2NET project, can be provided via excel files. Thus, this research work highlights a technique for processing the excel files in the LSHs. This technique adopts the concept of Multi-Agent Systems for processing the data as table in the excel files in the LSH. The multi-agent approach allows the LSH to process any excel file regardless the complexity in the data structure or in the file table. Furthermore, the presented approach enhances the processing of the excel files in different aspects, such as the size of the excel file or the required processing power.**

Keywords*— *Cloud Based; Data Collection; Enterprise; supply Chain; Multi-Agent Systems

I. Introduction

The collaboration between enterprises is currently a need and it is feasible due to the advances on the ICT field. However, this creates several challenges and, in turn, many opportunities and research objectives to merge the concept of collaborative network in an industrial manner. This is in the scope of the Industry 4.0, where the manufacturing system is partially or completely deployed as a cloud-based application. In this context, Internet of Things (IoT) enabled devices are required to collect data from the shopfloor of the factories. Besides, the legacy systems, such as ERP of companies need to be merged with new approaches. Similar to this, the C2NET project aims to provide a platform for addressing the supply chain interaction between customers, manufacturers and suppliers. The vision to employ the collaborative networks concept where companies are represented a network with several members that seamlessly communicate within the cloud-based platform. Furthermore, the C2NET tends to provide the targeted companies with optimization and monitoring features to increase the proficiency in terms of required communications.

As the C2NET targets the Small and Medium-sized Enterprises (SMEs), an adaptation of the legacy systems is expected. As an example, the MS excel file are intensively used in this category for enterprises since it is simple and well known for employees. This creates a challenge related to processing the excel files and extracting the data where it is going to be persisted in the databases of the C2NE platform. The C2NET architecture addressed this issue by proposing the concept of having Legacy System hubs (LSHs). These hubs are designed to bind the ERP data with the C2NET databases for further usage.

The rest of the article is structured as follows: Section II presents the literature review and industrial practices in the scope of this research work. Afterwards, Section III describes an approach for processing data of enterprises. Then, Section IV presents and discusses a proof of the concept that has been carried out in one pilot of the C2NET project. Finally, Section V concludes de paper.

II. State of the Art

A. *Multi-Agent Systems*

Many bibliographic sources provide formal definitions for agents in the scope of science and technology e.g., [1]–[3]. Fundamentally, an agent is an autonomous computer system that is capable of exchanging information with other peers throughout an agreed communication language. Then, a Multi-Agent System (MAS) consists on the deployment several agents that work collectively in order to perform tasks for specific objectives. In order to implement agents that participate in MAS, software engineers must face different design parts: agent design and society design [1]. Conceptually, meanwhile the capabilities of agents (i.e. agent design) are the ones that permit such entities to perform required tasks; the interaction of agents (i.e. society design) describes the behavior and rationale that agents follow in order to cooperate, coordinate and negotiate during agent-to-agent communication.

Commonly, ad hoc MAS are the ones being implemented when facing a problem. However, there are multiple frameworks and modelling software that can be used when implementing MAS. For example, The Foundation for

Intelligent Physical Agents (FIPA)[1] is organization that provide a set of specifications for implementing standardized agents. On the other hand, a comparison of existing MAS frameworks and tools can be found in [4], [5].

The employment of autonomous and intelligent MAS occurs in multiple domains and application types e.g., game theory, logistics, internet, health and industrial systems. Mainly, MAS support humans in order to perform tasks that are sometimes unsuitable. In the domain of manufacturing, MAS are commonly used for controlling distributed systems [6]–[8]. In addition, other applications in the field, such as integration, security or data processing are currently carried out by MAS. Depending on the field and type of application, MAS provide many benefits as, for example, autonomous behavior, dynamism, decision support, scalability and self-organization and configuration for different type of networks.

B. Industry 4.0

Industry 4.0 started as a national initiative in Germany, and quickly evolved to an extended concept used to identify the future of industrial production in which concepts as the Smart Factory fundamental [9]. Nowadays, it is visible as key area for research and development worldwide. e.g. European Commission programmes, such as the "Factories of the Future" (FoF), "Made in China 2025" or U.S. "Smart Manufacturing Leadership Coalition" (SMLC), all aiming the at the future and intelligent factory, flexible to the changing market demands, adaptable to the workers, and more agile concerning value chain integration.

Advances in digital automation can contribute significantly to change today's industrial sector where processes need to become less static. Together with flexibility and mobility, complexity is rising, representing an immense opportunity for technologies such as Cyber Physical Systems (CPS), Internet of Things (IoT), and Multi-Agent Systems (MAS). Indeed, the diminution of costs related to smart sensors and networks, and the development of advanced embedded systems have contributed for the rapid expansion of IoT-enabled factories - Industrial Internet of Things [10], [11].

CPS are defined as transformative technologies for managing interconnected systems between its physical assets and computational capabilities [12]. They act independently, co-operatively or as "systems-of-systems" composed of interconnected autonomous systems or agents originally developed to fulfil dedicated tasks [13]. New architectures such as the CPS 5C-level architecture [14] or OSMOSE [15] sensing enterprise are being created to provide advanced connectivity that ensures real-time data acquisition from the physical world and information feedback from the cyber space. Such real-time access to sensorial information together with advanced networking and processing capabilities provide a sort of 'nervous system' within the factory that brings Industry 4.0 closer.

CPS together with the Industrial Internet of Things enable the creation of a smart network of machines, ICT systems, smart products and people across the entire smart factory. This integration improves efficiency, optimizing resource planning, scheduling, and control. In terms of research, the work performed by Zuehlke (2010) [16] is worth to mention, but different smart factory implementations can already be found in industry as well (e.g. Ubisense Smart Factory[2]).

C. Manufacturing Systems and Supply Chain

A manufacturing system is any system that organizes and executes processes for creating products as, for example, automotive assembly lines. In order to organize all involved tasks that are carried out in manufacturing process, a large hierarchy of systems must be designed, implemented and maintained. The automation pyramid sketched in the well-known ISA-95[3] standard is the one that presents the structure of manufacturing enterprises [17]. This representation depicts three major levels: Enterprise Resource Planning (ERP), Manufacturing Execution System (MES) and shop floor which, in turn, consist of three types of control i.e., Batch, Continuous and Discrete control. Depending on the level of execution, the manufacturing system must handle physical or cyber operations performed at shop floor or MES, respectively. Then, it is obvious that the systems that perform such operations will be of different kind.

On the other hand, the supply chain is a system composed by resources that are involved in the manufacturing and delivery of products to customers. Then, the supply chain is a large system because it encompasses not only organizations, but also any type of asset that is employed for processing and shipping parts or finished products, from suppliers to customers. This is the reason why the supply chain includes different kind of things, such as people, processes, data or raw material [18].

Principally, the ERP level is the system that enables part of the integration of manufacturing enterprises with other supply chain resources. ERP unite suppliers, manufacturer and customers, which are the main actors of the supply chain. The management of the chain will be critical in order to reach efficiency of performance. Moreover, one of the major concerns when integrating manufacturing enterprises to other organizations belonging to the same supply chain, is an efficient manner of exchanging information [19]. This is currently being improved within complex ICT-based solutions that facilitates the communication of partners. The complexity of such solutions relies on the nature of the sector, which is highly dynamic and it includes multiple heterogeneous data sources.

D. Data in Enterprises

In current global economy, collaboration has become a key factor to deal with market changes, allowing enterprises to increase their competitive advantages, flexibility and agility [20][21][22]. In this regards, it is crucial that the enterprises legacy systems can communicate and exchange information [23]. Moreover, the changing conditions of the environment make enterprises to require innovative and speedy solutions to manage high amounts of data. Large companies as well as small

[1] http://www.fipa.org/
[2] http://ubisense.net/en/products/smart-factory
[3] https://www.isa.org/isa95/

and medium-sized companies normally have implemented different enterprise applications and software such as Enterprise Resource Planning (ERP) [24] or Customer Relationship Management (CRM) to manage their daily operations. Billions of information transactions are performed everyday by enterprises and the data exchanged is packaged in different formats. Enterprises need to exchange the data generated by these applications to other entities of their supply chain.

Different technologies are used by the enterprises to transfer business documents, such as Electronic Data Interchange (EDI) [25]; and manage information exchanges, including RFID [26]. The most common files exchanged by enterprises vary from text data (.txt) to excel files (.xls). Recent advances in Information and Communication Technologies (ICTs) lead to the use of cloud-based data exchange technologies, that allow to realize a high degree of performance, reliability, scalability and stability [27] . The Cloud Collaborative Manufacturing Networks (C2NET) H2020 European Project goes in this direction and amongst all its modules it is remarkable the Data Collection Framework (C2NET DCF) module, that provides software components and hardware devices for IoT-based continuous data collection from supply network resources [28].

III. Approach

As the C2NET project aims to provide a cloud-based platform for ERP supply chain interactions, collecting the companies' ERP data is basic requirement. Thus, the architecture of the C2NET platform includes a separate framework which so called Data Collection Framework (DCF). The DCF represents the gateway for ERP data to be transferred and stored in the platform for further exploitation such as optimization of production, delivery and logistic plans. With the support of the Legacy System Hubs (LSH), the data is transferred from the ERP to the DCF. One of the LSHs that implemented for binding the ERP with DCF is based on the PlantCockpit project[4]. The PlcantCockpit project provided an open source framework for service-based applications. The concept includes the approach of Function Blocks, which described in the IEC-61499 standard[5]. In fact, the PlantCockpit project targeted the control and monitoring factory facilities. However, for the C2NET project, the PlantCockpit has be evolved as a LSH.

A. Requirements

The LSH represents the binding layer between the company ERP and the DCF in the C2NET platform. Therefore, the LSH is required to follow the architecture requirements of the DCF in the C2NET. As shown in Fig. 1, the DCF contains the Resource Manager (RM) that organizes the data resources and allows the users to configure and visualizes the different data resources. As well, the RM receives the data form the resources and pass it to the Data and Knowledge Management System (DKMS). Besides storing the data of the resources, the DKMS maps and transforms the received data from the data sources to the pre-defined schema. This schema is called STables. The DKMS includes Data Identification and Transformation (DIT) module, which manages the mapping and transforming the received data to the STables schema, Low Level Complex Event Processing (LLCEP) module that allows the user to create events regarding to the received data and Knowledge–Base storage (KB) for storing the received data in the STables schema.

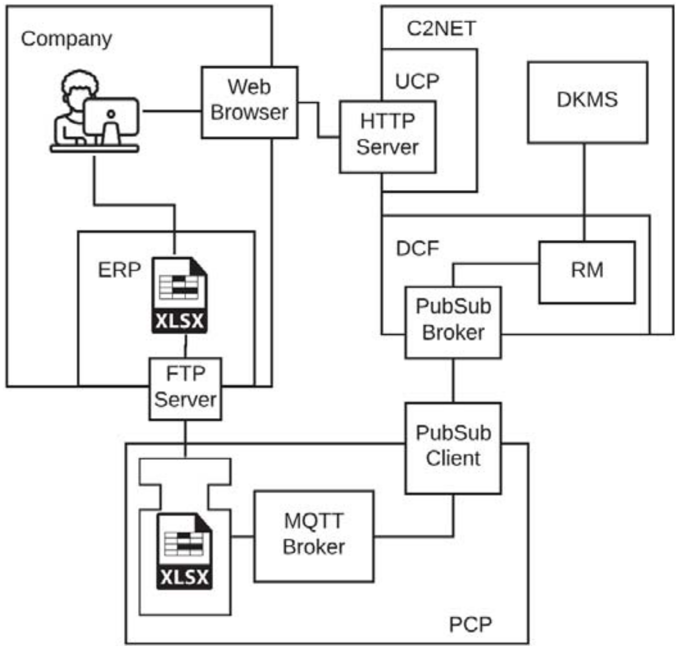

Fig. 1. C2NET, ERP and LSH components

On the other hand, the user on the company side provides the data as .xls/xlsx file. These files are accessible via FTP server. In addition, the user accesses the C2NET /DCF through a web browser for configuring the RM and related components. Finally, the LSH consists of PUB/SUB client for connecting to the DCF using the secured web socket standard on one side, and MQTT broker for connecting the LSH components and the function blocks for processing the .xls/.xlsx files on the other side.

According to that, firstly, the user is required to create the STables schema that satisfies the received data. Then, creates the mapping rules for the transformation process. After that, creates the CEP rules if needed. Finally, configure the LSH in the RM. All these requirement form the user side should involves a proper interface to ease the process for the user.

This research work focuses on how the LSH processes the .xls/.xlsx files. Since these files don not provide a standard schema, the configuration of the hub needs to be generic and flexible for adapting any .xls/.xlsx files. For this reason, a set of rules are established for achieving the mentioned requirements. These rules are:

- The LSH allows the user to select the required fields from the .xls/.xlsx files. The LSH sends these fields to the RM then to the DIT for mapping needs before sending the data.

[4] http://www.tut.fi/plantcockpit-os/

[5] http://www.iec61499.de/

- Each .xls/.xlsx file is considered as a data resource in the RM.
- The LSH sends the retrieved data from .xls/.xlsx file as an array of JSON object. Each JSON object includes one key-value of the selected fields.
- The LSH sends the data in String datatype. This means, the DKMS will transform the data type according to the predefined STables.

B. Design of Multi-agent System

As mentioned in the previous section, the genericity of the .xls/.xlsx files creates a challenge for the LSH to read and process these files. Besides that, as a requirement in the DCF, the LSH should provide list of the selected fields so the user can configure the mapping of the data before start sending the data.

On the other hand, the PlantCockpit requires the user to configure the IEC-61499 excel adapter function block. In this context, the logic behind any function block should be linked with incoming/out coming events. According to that, Fig. 2 depicts the structure of the excel adapter function block in the PlantCockpit LSH.

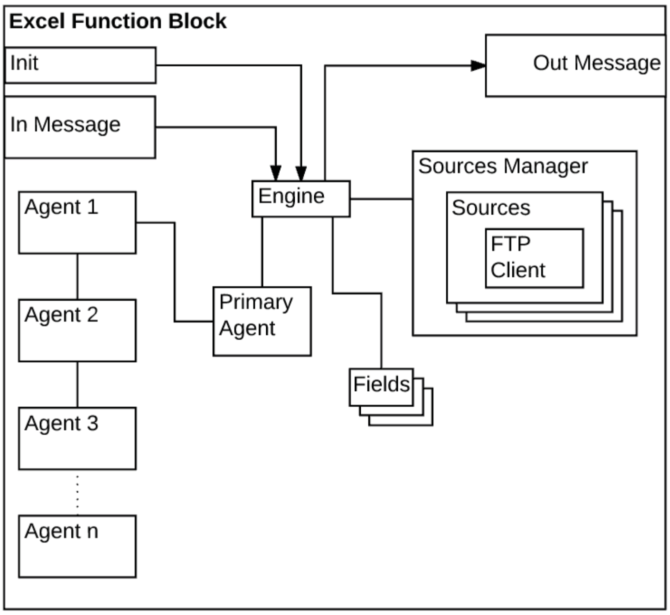

Fig. 2. Excel function block architecture

As shown in the figure, the excel function block accepts two types of event or messages; *Init* for initialization the function block components and *In Message* for passing incoming messages from other components. For e.g. the PubSub client. On the other side, the excel adapter function block sends messages through Out Message. In fact, both the *In Message* and *Out Message* are topics in the MQTT broker in the PlantCockpit architecture.

As well, the function block contains a source manager (SM). The SM manages more than one data source if needed. However, since the DCF already contains the RM, then, the SM will be managing just one source. Inside the SM, the *Source* is defined for managing and connecting to the data source. It includes an FTP client because the ERP excel files are provided via FTP server. In this manner, the *Source* is responsible for accessing, reading and extracting data from the excel files. The Agents, on the other hand, represents ranges in the excel sheet. As known, a cell represents the main structure of any excel sheet, each cell is addressed via rows (1, 2, 3...) and columns (A, B, C…). Besides, a set of cells are represented as a range in the excel sheet. These agents scan (loops) each cell in the range. During that, the agent updates its own current address. In this manner, there are two types of agents; Primary Agent and normal Agent. The Primary Agent is the only agent, which interacts with the *Engine*. Besides, the Primary Agent loop depends on the end of the data of the excel sheet. In other words, it scans the range until last row or column.

Another pillar in the architecture of the excel function block is the *Field*. It is responsible for calculating the address of the selected field by the user with respect to the current addresses to the available agents. Finally, the *Engine* component, which represents the core of the function block. It manages the data retrieval from the data source, handles the in/out messages, maps the selected fields and triggers the primary range for starting the retrieval process.

Fig. 3 shows the architecture of the primary agent and the normal agents. The *id* attribute has type string to provide unique identifier for the agent instance. Then, the *from* attribute holds the information where the agent starts looping in the dedicated range. The *filter* attribute provides ability to examine the contents of the cell using Regex expressions. Meanwhile, the *Step* defines the step value that the agent will increase each iteration. *ingnoreEmpty* allows the agent to ignore empty cells. Then, *isLast* attribute marks if the agent is the last agent in the chain. Finally, *nextAgent* is the instance of the next agent. As appears, the only difference between these agents is the last cell in the range and the direction. For the primary range, the Boolean attribute *isVertical* marks the direction if it is horizontal or vertical. Since the primary range scans all excel sheet then it finishes once the data in the sheet is ended. On the other hand, the normal ranges includes an attribute for marking the end cell address, with *to* alias for scanning the range as shown in the Fig. 3.

Primary Agent		
attributes	id	string
	from	string
	filter	string
	step	integer
	ignoreEmpty	boolean
	isVertical	boolean
	isLast	boolean
	nextAgent	agent
methods	init	void
	stepIn	json object
	reset	void
	isFinished	boolean
	getCurrentCell	string

Agent		
attributes	id	string
	from	string
	to	string
	filter	string
	step	integer
	ignoreEmpty	boolean
	isLast	boolean
	nextAgent	agent
methods	init	void
	stepIn	json object
	reset	void
	isFinished	boolean
	getCurrentCell	string

Fig. 3. Structure of the primary agent and the normal agents

For the methods, both primary range and normal range contain the same methods. In this manner, the *init* method initials the agent instance. Where, the *stepIn* forces the agent to increase the current cell address by the specified step. Meanwhile, the *reset* forces the agent to reset to the initial address. *isFinished* returns true if the agent reached the last address. Finally, the *getCurrentCell* returns the current cell address that the agent in.

The concept of Agents allows the flexibility in terms of processing the excel files. Fig. 4 shows the sequence diagrams of the interactions between the agents. It important to mention that the agents are defined as a chain starting with the primary range. The number of the normal ranges depends on the configuration that the user inserts that satisfies the structure of the excel table. This configuration decides the number of the agents in the chain. Each range is represented as an agent. The primary agent starts the scanning once it receives the *stepIn* call coming from the *Engine*. If the primary agent is the last agent in the chain, it increment the current address and return it back. On the other hand, if the primary agent is not the last agent in the chain, it calls *stepIn* method in the next agent. Then, once the primary agent receives the returned value, it adds its own current address to the returned object then, it calls the isFinished method in the next agent. If the next agent is finished, then the primary range calls *reset* method in the next range and increments its address. Finally, the primary agent returns the object that includes the current address of all agents in the chain. These all actions are repeated for all agents since they are representing a chain of agents.

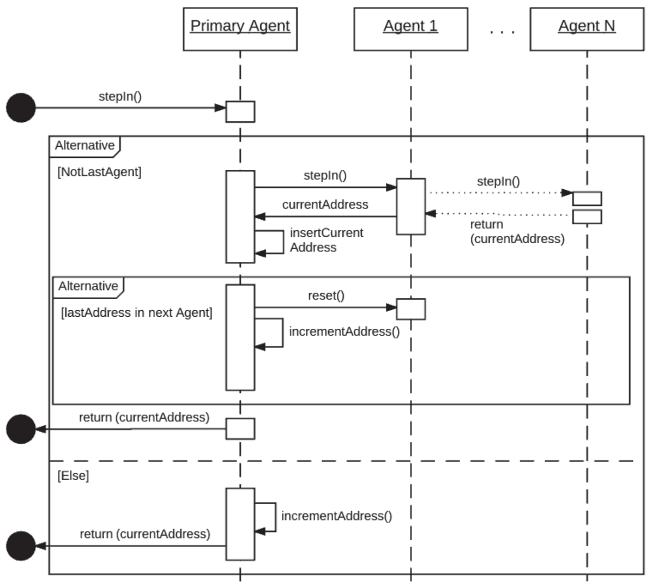

Fig. 4. Agents Interactions

In the Engine side, as appears in Fig. 5, the process starts by calling by *run* in the *Engine*. This triggers the *Engine* to call *stepIn* method in the primary range. As aforementioned, the primary agent returns all current addresses of all agents in the chain. In this context, and according to the selected fields, the engine start reading the excel sheet and collect the values according to the mapping between the selected fields and the agents. This whole action continues until the primary agent reaches the end of its range. In this matter, the excel sheet is opened just once and the *Engine* that reads specific cells which reduces the processing power that is needed for such a task. As well. This approach allows the user to process any excel file since the concept maps ranges to agents then to fields.

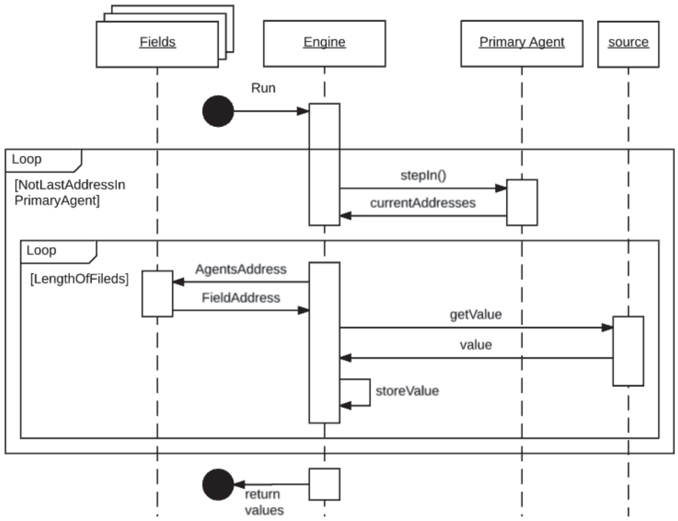

Fig. 5. Generating output value

IV. IMPLENETATOIN AND DISCUSSION

In the C2NET project, this approach is used in one of the defined pilots, which addresses the interactions between a supplier and manufacturer related to automotive manufacturing. The LSH is used to retrieve the data from the ERP of manufacturer, which includes production plans for the manufacturer, in order to be optimized in the C2NET project. This section presents the implementation of the presented approach. As well, it shows the results that are achieved using the use-case.

A. Automotive manunfacturing use-case

In order to validate C2NET results, different industrial validation scenarios have been defined. One of them involves a first and a second-tier supplier of an Original Equipment Manufacturer (OEM) of the automotive industry sector. The validation scenario is related to the optimisation of a Collaborative Materials Requirement Plan among the first and the second-tier supplier. The first-tier supplier receives the demand from the OEM and launches the optimization algorithm in order to calculate the needed materials. This optimization minimizes its order and inventory cost. After validating the result, the second-tier supplier is notified that a new optimized plan from its customer (the first-tier supplier) has been released. This optimized plan is the input data for the second-tier supplier to calculate its own materials requirement plan. Depending on the results, both companies could start a negotiation process in order to agree on the dates and quantities of materials to be delivered.

The example of this research work focuses on the first part that involves the data transaction in which the first-tier supplier receives the demand plan from the OEM. This demand plan is constructed as an excel file. TABLE I. shows the structure of the plan in an excel format.

TABLE I. DATA STRUCTURE OF THE DEMAND EXCEL FILE

◢	A	J	K	…	BJ
1	**Material**	**Period_1**	**Period_2**	**…**	**Period_53**
2	Material_1	Demand_11	Demand_12	…	Demand_153
3	Material_2	Demand_21	Demand_22	…	Demand_253
…	…	…	…	…	…
n	Material_n	Demand_n1	Demand_n2	…	Demand_n53

In order to launch the optimisation, the data of this source should be transferred to the C2NET database. The Standardised Tables (STables) are part of the C2NET database, providing common vocabularies for all the involved users. STables offer to C2NET the required structured and standardised features for data collection, optimisation, and collaboration purposes. Accordingly, C2NET data is the standardised and homogenised data used in the Cloud Platform.

In the studied validation scenario, each material code (Material_n) is associated with the C2NET materials' identifiers (PartID) that are already defined in the C2NET database. In the same way, each of the periods' codes (Period_m) are linked to the already defined identifiers of the periods in the C2NET database (PeriodID). These periods in this example represent a week. Thus, in the table a 53 period or week is found. This means that this plan covers one year. Once this matching has been performed between the PartID and the PeriodID, the demand data (Demand_nm) is loaded to the corresponding field RequirementAmount, of the STable Part_Period. It is important to mention that the excel file contains more data about the materials. However, for this implementation, data are not presented.

B. *Results*

As previously mentioned, the configuration of the hub plays a role in the agent's chain construction. In other words, the user configures the LSH according to the excel file structure. Then, the *Engine* in the LSH creates the agents according to the configuration. Each defined range in the configuration is represented as an agent. Fig. 6 shows the configuration that is needed to retrieve the required data for the aforementioned use-case. As shown, two agents are needed for handling the complexity of the excel file. The primary agent loops in column A, where also, the material data is located. This means that this agent will scan until last row in the table. On the other hand, the second agent covers the week's row (periods), which starts from J1 until BJ1. Besides the ranges, three Fields are selected which includes the material, the week as a period and the demand. As shown in the figure, the fields are mapped to the ranges via the row and column dependencies. The value points to the id of the defined ranges. As an example, the material field will follow the primary range without any offset. Meanwhile, the demand depends on the primary range in the row address and week in the column address. As well, the offset value allows the filed to be retrieved event though; it is not on the exact range. For this example, none of the fields has any offset.

```
{
  PrimaryRange: {
    id: "1",
    from: "A2",
    filter: "*",
    step: 1,
    ignoreEmpty: true,
    isVertical: true,
    sheet: "Hoja1"
  },
  embeddedRanges: [
    {
      id: "2",
      from: "J1",
      to: "BJ1",
      filter: "*",
      step: 1,
      ignoreEmpty: true,
      sheet: "Hoja1"
    }
  ],
  fields: [
    {
      name: "Hoja1.Material",
      rowDependency: "1",
      columnDependency: "1",
      rowOffset: 0,
      columnOffset: 0,
      sheet: "Hoja1"
    },{
      name: "Hoja1.week",
      rowDependency: "2",
      columnDependency: "2",
      rowOffset: 0,
      columnOffset: 0,
      sheet: "Hoja1"
    },{
      name: "Hoja1.demand",
      rowDependency: "1",
      columnDependency: "2",
      rowOffset: 0,
      columnOffset: 0,
      sheet: "Hoja1"
    }
  ]
}
```

Fig. 6. The multi-agent system configuration with respect to the use-case

For simplifying the interaction with the C2NET, the user is provided with a proper interface for configuring the LSH. Fig. 7 depicts a screenshot of the configuration interface, where the user firstly, selects the fields that are required. Then the user is entitled to add the ranges. Finally, the user maps the ranges with the selected fields.

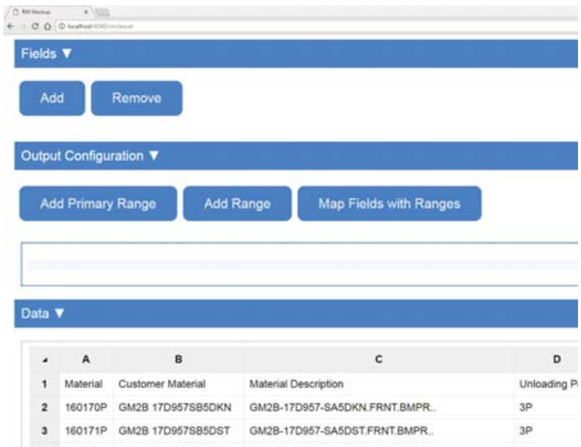

Fig. 7. LSH configuration interface

The results of this research work approach using the use-case is shown in Fig. 8 where the LSH extracted the data from the excel file and send it as a JSON array. As depicted, each selected field is presented in a JSON object. This object is an element of the total JSON array.


```
[
  {
    "Hoja1.demand" : "3",
    "Hoja1.week" : "442,016",
    "Hoja1.Material" : "160171P"
  },{
    "Hoja1.demand" : "13",
    "Hoja1.week" : "442,016",
    "Hoja1.Material" : "160173P"
  },{
    "Hoja1.demand" : "5",
    "Hoja1.week" : "442,016",
    "Hoja1.Material" : "160174P"
  },{
    "Hoja1.demand" : "6",
    "Hoja1.week" : "442,016",
    "Hoja1.Material" : "160175P"
  },...
]
```


Fig. 8. The results that extracted from the use-case ERP

V. Conclusion

The presented approach adopted the multi-agent system concept for processing the excel files. Mainly, each agent represented a range in the excel file. This provided a generic approach where the complexity of the structure of the excel tables can be handled. As well, the user is able to select the fields that are needed to be extracted. Moreover, the presented approach showed an enhancement in the performance when regarding the required processing power and the management of large sized files.

Although aforementioned characteristics show the potential of presented approach, the architecture could be enhanced with functionality for allowing several primary ranges. This will be designed and implemented in further work. Therefore, the LSH will be capable of processing multi-dimensional excel files.

Acknowledgment

The research leading to these results has received funding from the European Union's Horizon 2020 research and innovation program under grant agreement n° 636909, correspondent to the project shortly entitled C2NET, Cloud Collaborative Manufacturing Networks.

References

[1] W. Michael, *An Introduction to MultiAgent Systems*, Second edition. John Wiley & Sons, 2009.

[2] S. J. Russell and P. Norvig, *Artificial Intelligence: A Modern Approach*. Prentice Hall, 2010.

[3] F. Jaques, *Multi-Agent Systems: An Introduction to Distributed Artificial Intelligence*. Addison-Wesley Longman, 1999.

[4] R. J. Allan, "Survey of Agent Based Modelling and Simulation Tools," Report DL-TR-2010-007, 2009.

[5] C. N. and G. Madey, "Tools of the Trade: A Survey of Various Agent Based Modeling Platforms," 31-Mar-2009. [Online]. Available: http://jasss.soc.surrey.ac.uk/12/2/2.html. [Accessed: 21-Apr-2017].

[6] V. V. Herrera, A. Bepperling, A. Lobov, H. Smit, A. W. Colombo, and J. Lastra, "Integration of Multi-Agent Systems and Service-Oriented Architecture for industrial automation," in *6th IEEE International Conference on Industrial Informatics, 2008. INDIN 2008*, 2008, pp. 768–773.

[7] P. Leitao, S. Karnouskos, L. Ribeiro, J. Lee, T. Strasser, and A. W. Colombo, "Smart Agents in Industrial Cyber–Physical Systems," *Proc. IEEE*, vol. 104, no. 5, pp. 1086–1101, May 2016.

[8] B. Vogel-Heuser, C. Diedrich, D. Pantförder, and P. Göhner, "Coupling heterogeneous production systems by a multi-agent based cyber-physical production system," in *2014 12th IEEE International Conference on Industrial Informatics (INDIN)*, 2014, pp. 713–719.

[9] S. R. C., K. M., and M. P., "Industry 4.0 Challenges and Solutions for the digital transformation and use of exponential technologies," 2015, p. 32, Deloitte AG.

[10] M. Marques, C. Agostinho, G. Zacharewicz, and R. Jardim-Gonçalves, "Decentralized decision support for intelligent manufacturing in Industry 4.0," *J. Ambient Intell. Smart Environ.*, vol. 9, no. 3, pp. 299–313, Jan. 2017.

[11] B. Ramis Ferrer and J. L. Martinez Lastra, "Private local automation clouds built by CPS: Potential and challenges for distributed reasoning," *Adv. Eng. Inform.*, vol. 32, pp. 113–125, Apr. 2017.

[12] R. Baheti and H. Gill, "Cyber-physical systems. The Impact of Control Technology," *Impact Control Technol.*, vol. 12, pp. 161–166, 2011.

[13] P. Guturu and B. Bhargava, "Cyber-physical systems: a confluence of cutting edge technological streams," in *International conference on advances in computing and communication*, 2011, vol. 138.

[14] J. Lee, B. Bagheri, and H.-A. Kao, "A Cyber-Physical Systems architecture for Industry 4.0-based manufacturing systems," *Manuf. Lett.*, vol. 3, pp. 18–23, Jan. 2015.

[15] C. Agostinho *et al.*, "Osmosis Process Development for Innovative Product Design and Validation," p. V02BT02A027, Nov. 2015.

[16] D. Zuehlke, "SmartFactory—Towards a factory-of-things," *Annu. Rev. Control*, vol. 34, no. 1, pp. 129–138, Apr. 2010.

[17] MESA International, "MESA White Paper #39: MESA Model Evolution," 2011.

[18] P. Rolf G., "Application of the SCOR Model in Supply Chain Management By Rolf G. Poluha." [Online]. Available: http://www.cambriapress.com/cambriapress.cfm?template=4&bid=37. [Accessed: 21-Apr-2017].

[19] R.P. Kampstra, J. Ashayeri, and J.L. Gattorna, "Realities of supply chain collaboration," *Int. J. Logist. Manag.*, vol. 17, no. 3, pp. 312–330, Sep. 2006.

[20] L. M. Camarinha-Matos and H. Afsarmanesh, "Collaborative networks : a new scientific discipline," pp. 439–452, 2005.

[21] B. Andres and R. Poler, "Models , guidelines and tools for the integration of collaborative processes in non-hierarchical manufacturing networks : a review," *Int. J. Comput. Integr. Manuf.*, 2015.

[22] M. Lauras, J. Lamothe, F. Benaben, B. Andres, and R. Poler, "Towards an Agile and Collaborative Platform for Managing Supply Chain Uncertainties," *IFIP Int. Fed. Inf. Process.*, vol. 1, pp. 64–72, 2015.

[23] J. L. Seng and Z. Wong, "An intelligent XML-based multidimensional data cube exchange," *Expert Syst. Appl.*, vol. 39, no. 8, pp. 7371–7390, 2012.

[24] A. Boza, L. Cuenca, R. Poler, and Z. Michaelides, "The interoperability force in the ERP field," *Enterp. Inf. Syst.*, vol. 9, no. 3, pp. 257–278, 2015.

[25] B. Jardini, M. El Kyal, and M. Amri, "The complexity of Electronic Data Interchange (EDI) compliance for automotive supply chain," *IEEE Int. Conf. Ind. Eng. Eng. Manag.*, pp. 361–365, 2016.

[26] B. Bindi, R. Bandinelli, and R. Rinaldi, "RFID and eBIZ implementation in the Textile and Clothing Industry: Evidence of a pilot project," *Proc. Summer Sch. Francesco Turco*, vol. 13–15, pp. 156–160, 2016.

[27] Y. Hao, P. Helo, and A. Shamsuzzoha, "Cloud-based data exchange and messaging platform implementation for Virtual Factory environment," *IEEE Int. Conf. Ind. Eng. Eng. Manag.*, pp. 426–430, 2016.

[28] B. Andres, R. Sanchis, and R. Poler, "A Cloud Platform to support Collaboration in Supply Networks," *Int. J. Prod. Manag. Eng.*, vol. 4, no. 1, pp. 5–13, 2016.