Legacy Systems Interactions with the Supply Chain Through the C2NET Cloud-based Platform

Khurshid Ali Qureshi, Wael M. Mohammed, Borja Ramis Ferrer, Jose L. Martinez Lastra
Tampere University of Technology
FAST Lab, TUT
Tampere, Finland
{khurshid.qureshi@student.tut.fi, wael.mohammed@tut.fi, borja.ramisferrer@tut.fi, jose.lastra@tut.fi}

Carlos Agostinho
Centre of Technology and Systems
CTS, UNINOVA
2829-516 Caparica, *Portugal*
{ca@uninova.pt}

***Abstract*— Enterprise Resource Planning (ERP) represents the highest level of the manufacturing systems according to ISA-95 standard. It includes supply chain management, and customer-manufacturer-supplier interactions, essential to a successful business relationship in the frame of Industry 4.0. Many research efforts are being put in place to follows the latest ICT advances, combining them with manufacturing systems technologies. In this manner, the C2NET factories of future project (c2net-project.eu) has been put in place to provide a cloud-based collaborative network for supply chain interactions. This research paper focuses on providing a solution for retrieving ERP data via a Legacy System Hub, which is powered by the previous PlantCockpit project (www.tut.fi/plantcockpit-os/) outcomes. This solution allows the C2NET users to transfer the ERP data from the company side to the C2NET platform via the Data Collection Framework (DCF). Thus, the contribution of this research work is a generic solution that enables the interaction between systems owned by different organizations involved in the same supply chain. As C2NET targets the Small and Medium-sized Enterprises (SMEs), three data adapters are developed for MS Excel, SQL and RESTful data sources. The result of this research enhances the users experience in terms of interface and it might help in the C2NET databases for controlling and monitoring the instance changes in the ERP data on the company side.**

Keywords—Industry 4.0, data collection, supply chain, Enterprises, SME

I. Introduction

Current industrial enterprises employ new IT-based solutions for increasing their efficiency and connectivity of their systems. However, companies must confront interoperability issues because their legacy systems are not always capable to be integrated with contemporary systems e.g., due to technological gaps. Frequently, companies struggle in order to maintain legacy systems but they eventually i) succumb to replace them for new ones or ii) develop temporary ad hoc solutions.

Furthermore, the productivity of Small and Medium-sized Enterprises (SMEs) depend on the convenience for their users to access the provided services. In the scope of the Industry 4.0 [1], one of the approaches is to involve SMEs in collaborative networks; aiming the increment of their service visibility. In this way, SMEs can be incorporated to a common ecosystem wherein other organizations working in the same value chain can exchange information with no need of investing in IT infrastructure [2].

In order to permit the collaboration and optimization of processes, which are executed through the industrial supply chain, the Cloud Collaborative Manufacturing Networks (C2NET) project[1] is developing a portable and interoperable cloud-based platform [3]. Interaction of different organizations working in the same value chain is made feasible within the C2NET platform, and one of the challenges of the cloud-based platform is the collection of data. Information that is received from heterogeneous data sources must be integrated, transformed and served as useful information to interested parties in the same value chain.

The contribution of this research work is a generic solution for enabling the interactions of supply chain systems with interconnected companies and their legacy systems. As described along the paper, this research work addresses aforementioned objective mainly throughout i) the C2NET cloud-based platform, ii) the interaction between publisher (server) and subscriber (client) approach and iii) the employment of a function block based solution. It should be noted that this research goes beyond the previous work presented in [2], which offered a proof of concept for collecting data within PLANTCockpit-OS and the implementation of a legacy systems hub (LSH).

The rest of this manuscript is structured as follows: Section II presents the research background, which is mainly focused on i) industrial legacy systems, ii) supply chain implementation for the Industry 4.0 and iii) the C2NET data collection framework. Then, Section III describes the approach for enabling the interactions of legacy systems throughout the supply chain. This is presented within a description of the solution components and their interrelationships. Afterwards, Section IV demonstrates how the approach is implemented in two scenarios, based on two

[1] http://c2net-project.eu/

different adapters: Excel and REST. Finally, the paper is concluded in Section VI.

II. Background

A. Industrial Legacy Systems to ERP

Enterprise Resource Planning (ERP) is a set of systems having multiple features and transactional dimensions [1]. ERP improves the business process and enhances the efficiency of enterprise operations [1], [2] within the integration of information into a single unit. Currently, ERP systems are connected with supply chain and customer relationship management. Such interconnection permits to overcome the challenges of the legacy system [3].

A legacy system is one that resist changes, such as updating and adapting to the latest technological environment. Nevertheless, it is vital to business and its cessation can result into immense loss [4]. Legacy systems have large unstructured source codes, which are not properly documented. In fact, this makes their modification difficult [5]. Moreover, legacy systems have huge cost of maintenance and upgradation.

With the advancement in technology, companies require an efficient, flexible and scalable system, which can meet the growing demands of business more adequately. Unfortunately, legacy systems are unable to cope up the changing business needs due to their structural rigidity. Furthermore, the replacement of legacy system is risky, costly and time intensive. Thus, one of the most proficient ways to deal with the problems of legacy systems is their integration with modern automation systems. W. Ulrich defines the process of modernization as the one in which a new legacy system is evolved to meet the growing business requirements [6]. However, the legacy system integration is also a complex process because their monolithic structure is hardly compatible with any other software and hardware [4]. Moreover, previous work on the integration of legacy systems in the supply chain provides a summary on available integration systems [7]. More precisely, it presents different type of approaches for integrating legacy systems, such as agent based wrapper mechanism [8], a business process reengineering [9] or a role based access control [10].

B. Supply Chain in Industry 4.0

Supply chain management is about the timely, efficient and cost effective delivery of goods, services, information and finances among customers, suppliers, retailers and distributors. Since long time, the supply chain industry has been dealing with operational, financial and demand forecasting challenges.

The integration of cloud network in supply chain industry is a new global concept, which has revolutionized the supply chain management. Cloud computing is a concept of IT industry comprising of both software and hardware. It allows the user to get the services on demand without the involvement of equipment and location [10]. In a supply chain industry, cloud computing links two or more users through cloud services by sharing information and resources [11].

Using cloud based solution, companies are able to reduce operational cost because unlike traditional legacy ERP they do not come up with maintenance and upgradation cost [12]. As cloud computing offers a single platform for information sharing in which members can be easily added, it mitigates the complexity of supply chain. Further, it assists companies to forgo agility problems, as cloud platforms are not bound of location. Hence, cloud computing provides real time visibility of supply chain to foresee any gaps and take action promptly.

C. Data Collection Framework in C2NET

As explained before, the way that manufacturing and service industries manage their businesses is changing due to emerging cloud environments. Indeed, the high-performance and quality needs, together with cost-effective and flexible productivity is one of the emerging research challenges in the domain of Industry 4.0. Sensing enterprise is one of the paradigms to enable that, envisioning the enterprise as a smart complex entity capable of sensing and reacting to real time stimuli [11]. Hence, it is necessary to address the multiple dimensions of big data [12], capturing value-added data and making it available for data intensive environments to analyze, curate and ultimately provide novel services out of it.

The C2NET Data Collection Framework (DCF) is a solution being developed to address the first dimensions, providing software components and hardware devices for continuous data collection from manufacturing supply network resources to a cloud environment that enables highly scalable, available and fault-tolerant services. This DCF is composed of two main components: one to be deployed at the company premises, i.e., the Company middleware (CM), responsible for handling the IoT and Legacy systems data communication; and the DCF Cloud that implements resource virtualization techniques and manages data transformation when necessary [13], [14].

In more detail, the CM connects the factory systems to the C2NET cloud component. Due the need to handle different types of incoming data from IoT and Legacy systems, the component is divided in two distinct sub-components, IoT and LS Hubs (IoTH and LSH). Based on PlantCockpit [7], the LSH simple data adapter unifies the data from the heterogeneous data source interfaces connected to the hub. The DCF Cloud modules are designed as a publisher/subscriber approach, hosted in a cloud-based service ecosystem that allows to register and manage resources (Resource Manager) and monitor data collected from each device connected to the hubs.

III. Approach

The need of a generic and independent solution for retrieving companies' ERP data is considered as a main objective of the C2NET project. In fact, the motivation of this research work is to provide an approach for fulfilling the requirements of the C2NET the DCF. One of the objectives addresses the variations of the data sources. For example, MS Excel files which are exploited widely by SMEs in the enterprise level. Since, the C2NET targets the SMEs, MS Excel files are strongly considered to be one of the sources that needs to be included in the solution of this research work. Besides, the presented solution evolves the SQL data sources over HTTP (Hypertext Transferring Protocol) and RESTful web services. This section includes a representation of the main components in the solution.

Subsequently, an illustration of the interactions between the solution components are elaborated.

A. Main Components

The solution of this research work tends to cover i) the variations of the data sources in the SMEs category, ii) provide continuous monitoring of the updates of the data in ERP of the company and iii) easy web-based interface for the user. Figure *1* depicts the LSH and the related main components of the C2NET/DCF.

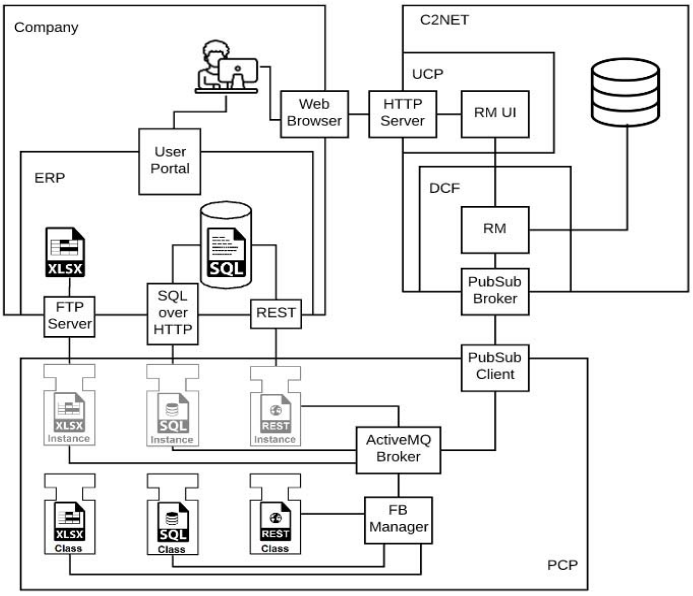

Figure 1. Main Components

As shown in Figure 1, three parties are included in the process of transferring of the company data to the C2NET platform. i) The ERP, which is the source of the data, ii) the LSH, and finally, iii) the C2NET DCF Cloud component, which receives the transported data for further use by the C2NET platform.

On the company side, the ERP system provides the company's data via several technologies. As expected, SMEs could provide their data using MS Excel files, SQL with query over HTTP and SQL with a RESTful Interface. These data sources are accessible by the user via a dedicated portal or manual tools. As well, the company user is able to configure the LSH using a web browser on the platform side.

According to Figure 1, the C2NET Platform contains four major components, which are involved in the data collection process. i) The Resource Manager (RM) for data sources management and configuration, ii) Resource Manager User Interface (RM UI) for visualizing and interacting with users, iii) HTTP server for hosting the required services and finally, iv) the Publish/Subscribe (PubSub) Broker for guaranteeing a tunnel for the hubs to exchange data and configuration.

Lastly, in Figure 1, the PlantCockpit LSH represents the bondage between the company and the C2NET DCF. The PubSub client assures the connection with the C2NET DCF Cloud. This connection allows the LSH and C2NET to exchange the data and the configurations that are needed for the data retrieval process. Apart from that, the Function Block Manager (FBM) creates and manages any function block class (FBCs) available in the LSH. In this research work, three classes are used; MS Excel, SQL and RESTful classes or on other words adapters. As seen in Figure 1, the function Block manager creates as many as required function block instances (FBI, highlighted in gray in the figure) out of the FBCs. Each instance will be connected with the suitable data source in the ERP system of the company. Finally, all internal components of the LSH are connected via an ActiveMQ broker, which uses Message Queue Telemetry Transport (MQTT) protocol.

Further details of the FBI can be seen in Figure 2. Each FBI registers two topics in the ActvieMQ broker; *inputTopic* for incoming messages and *outputTopic* for sending messages. These topics are used but the PubSub client for interacting with the FBIs. The engine is used for managing the data retrieval process. Meanwhile, the sources connect to the data source in the ERP. In this manner, the sources are managed by the sources manger. The sources manager provides the engine with set of methods for retrieving the data.

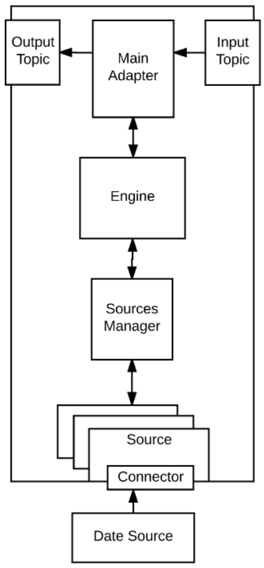

Figure 2. Structure of a Function Block Instance.

B. Components Interaction

The presented component in the previous subsection uses different technologies and techniques to communicate with each other. This sub section highlights the interaction between these components via sequence diagrams. Figure 3 is a sequence diagram that depicts the LSHub platform component interaction. As shown in the figure, the PubSub client receives the configuration fields from the PubSub server in the C2NET platform. Subsequently, it forms a complete XML configuration from those configuration fields. The XML configuration is shown in Figure 4.

This XML configuration is then forwarded to the FBM, which creates the FBI according to the received configuration. The newly created FBI manages its source and initiates the

Engine. Afterwards, it sends a notification back to the PubSub client about the availability of the FBI.

The PubSub client requests the FBI for the fields from the data source that the user needs to send to the DCF. After fetching the requested name fields from the data source, the FBI produces an HTML view of those fields and delivers them to the PubSub client. In the next step, PubSub client transfers it to the PubSub server located at the C2NET platform. At this point, the PubSub client receives an output configuration from the C2NET PubSub server. Later on, PubSub client parses the information and forwards the configuration to the FBI. Eventually, the FBI runs its Engine to acquire data from the data source and directs it back to the PubSub client.

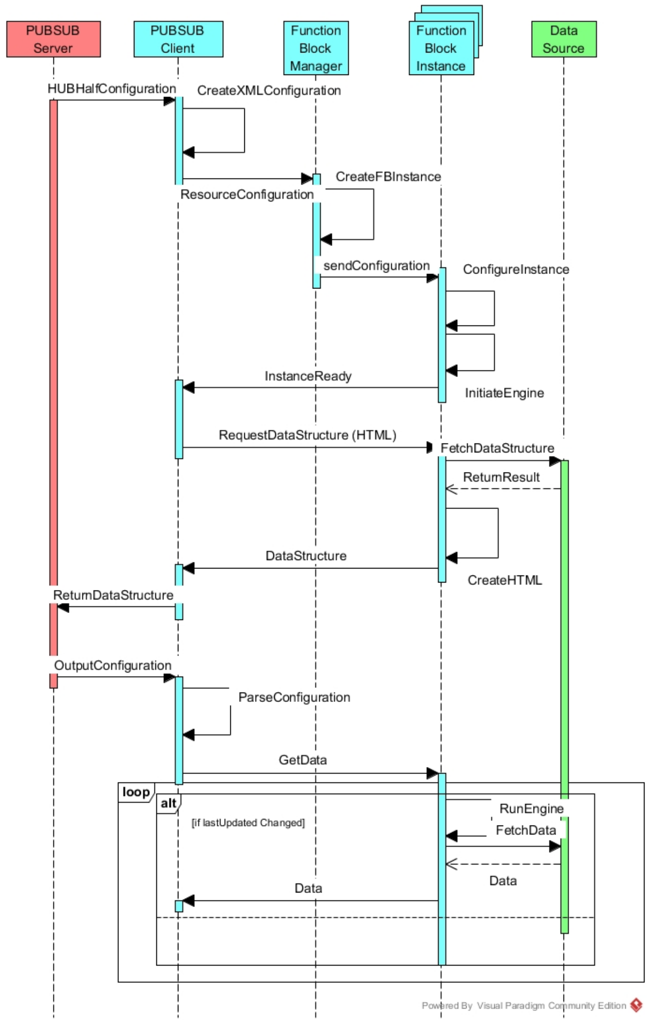

Figure 3. Sequence Diagram showing communication between the components of Legacy System Hub

The FBI examines the data source to identify any changes after every specified time interval. Hence, if there is any new change recorded in the data, the system is promptly updated. This is done by repeating the process of running the Instance Engine for fetching the data from the data source and sending it to the PubSub client. In this way, any revised data is securely obtained.


```xml
<?xml version="1.0" encoding="UTF-8"?>
<createFbInstances>
    <fbInstance>
        <fbId>test</fbId>
        <fbPid>pid_eu.plant.Testing_Block</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>
            </outputMessageType>
        </outputMessageTypes>
    </fbInstance>
</createFbInstances>
```


Figure 4. Input XML configuration to the FBM

In the correspondence of Figure 4, Figure 5 shows the component interaction of the C2NET platform. Here, the user initiates the process by entering the configuration fields in the RM. This configuration from RM is forwarded to the PubSub server, which then converts the received configuration into JSON configuration. Consequently, the RM transmits such configuration to the PubSub client on the LSHub side.

Next, the PubSub client returns an HTML view back to the PubSub server. Following, PubSub server sends this view to the RM, which displays the HTML on its user interface for configuring the data source. Then, the user is supposed to select fields from that interface and generates an output schema. At this point, RM passes that output schema to the PubSub server, which forwards such schema to the PubSub client in order to fetch the final data. This definite data is delivered back to PubSub server from PubSub client, which is ultimately transferred to The RM for the availability of the user.

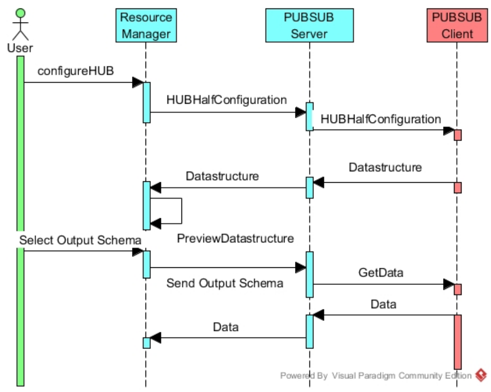

Figure 5. Sequence Diagram showing communication between the components of C2NET

IV. IMPLEMANTATION OF THE USE-CASES

After presenting the approach in the previous section, this section presents the usage of the presented approach in two use cases; MS excel and RESTful. Both use-cases follow the same presented approach. The difference is noticeable in the data source.

A. *Excel Adapter*

As illustrated previously, one of the data that needs to be retrieved from the ERP is the MS Excel files. Since the format of the sheets in the MS Excel files may vary, thus, to provide a generic form of configuration to the user, the following rules are formed:

- Each file represents a resource in the C2NET DCF.
- The LSH returns the data as array of object. Each represents a row of data.
- The LSH returns all data in String format. The transformation and parsing processes are occurred on the C2NET DCF side.
- An FTP server hosts the MS Excel files.

Once these rules are agreed, the LSH is able to provide the data from the MS Excel file. The configuration requires four exchange processes. Firstly, the RM sends the resource accessibility configuration or "*Half Configuration*" to the LSH. Figure 3 shows the configuration that the RM sends to the LSH. This configuration allows the FBI to access the FTP server.

```
{
    "path":"/ExampleSheets",
    "hostname":"192.168.1.176",
    "password":"",
    "name":"P1.xls",
    "id":"1",
    "type":"excel",
    "username":"c2net"
}
```

Figure 3. Data source configuration

Accordingly, the LSH downloads the excel file and creates an HTML script which contains the structure of the Excel file with some functionalities for choosing the required fields and configuring the LSH. See Figure 4. This HTML script is sent by the LSH to the RM.

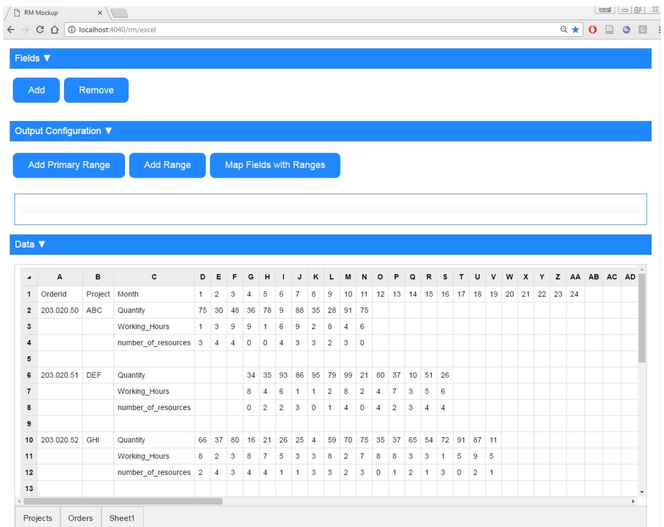

Figure 4. The visualization of the data resource in the RM

Then the user configures the Hub via the embedded HTML script. As shown in the figure, the user is required to choose the fields' names first. After that, the user configures the Hub according to the selected fields and the nature of the Excel file. Figure 5 shows the complete configuration.

```
{
    "PrimaryRange": {
        "id": "1",
        "from": "2",
        "filter": "*",
        "step": 1,
        "ignoreEmpty": false,
        "isVertical": true,
        "sheet": "Projects"
    },
    "embeddedRanges": [{
        "id": "2",
        "from": "D1",
        "to": "AA",
        "filter": "*",
        "step": 1,
        "ignoreEmpty": false,
        "sheet": "Projects"
    }],
    "fields": [{
        "name": "Orders.OrderId",
        "rowDependency": "1",
        "columnDependency": "1",
        "rowOffset": 0,
        "columnOffset": 0,
        "sheet": "Orders"
    },{
        "name": "Orders.Project",
        "rowDependency": "1",
        "columnDependency": "1",
        "rowOffset": 0,
        "columnOffset": 1,
        "sheet": "Orders"
    }]
}
```

Figure 5. Output schema configuration

The *PrimaryRange* represents the range in the Excel file that the LSH engine will loop until the end of the excel file entries. This range can be horizontal, which means, the hub will reach the last column in the excel file, or vertical which means, the LSH will loop to the last row in the excel file. The configuration accepts one *PrimaryRange*. Then the user defines the *EmbeddedRanges*, as seen in the figure, the embedded ranges are formatted as a JSON array. These ranges can occur as many as needed. Also, in the case of normal table (column wise or row wise). As well, the *EmbeddedRanges* should be bounded. This means the user should specify the start and the end addresses of the ranges.

Lastly, the user start mapping the selected fields and the configured ranges according to the location of the data with respect to the rages.

After the Hub is configured, it reads the Excel file and extracts the data according to the ranges and the selected fields in the configuration. The LSH output can be seen in Figure 6.

```
[
    {
        "Orders.OrderId":" 203.020.50",
        "Orders.Project":" ABC"
    },
    {
        "Orders.OrderId":" 203.020.51",
        "Orders.Project":" DEF"
    },
    …
]
```

Figure 6. A snapshot of the retrieved data

B. *RESTful Adapter*

The purpose of REST Adapter is to fetch data from REST HTTP sources, which in turn returns JSON object as a response. In this use case, this adapter is being used to fetch data from mock REST API, which individually return data from different data tables of a database in JSON format. The format of all the REST sources is a set containing data objects in an array as shown in the Figure 7:

```
[
    {
        "customerPurchaseOrder":"",
        "actualCost":"0"
    },
    {
        "customerPurchaseOrder":"null",
        "actualCost":"0"
    }
]
```

Figure 7. Example data from a REST Source

In addition to rules mentioned above for the Excel Adapter, following are the rules to form a generic configuration for the REST Adapter:

- Each REST Source represents a resource in the C2NET DCF.
- An HTTP server hosts the REST Sources.

The resource accessibility configuration RM sends in case of the REST Adapter is shown in the Figure 8:

```
{
    "id":"1",
    "url":"http://130.230.181.107:3001/merphelper/api/database/Order",
    "method":"GET",
    "fbpid":"RA1",
    "fbid":"pid_eu.plant.restAdapter"
}
```

Figure 8. Half Configuration for REST Adapter

Configuration comprises the details of the REST source, from which the data will be extracted. The attributes include the "*id*" of the source, its HTTP "*method*" and the "*url*" of the desired data, which is going to be collected. The LSH fetches the JSON data from the REST source and forms an HTML script, which represents the data in a tabular form. It also has some functionalities for choosing the required fields and configuring the LSH. Figure 9 shows the HTML view sent by the LSH to the RM.

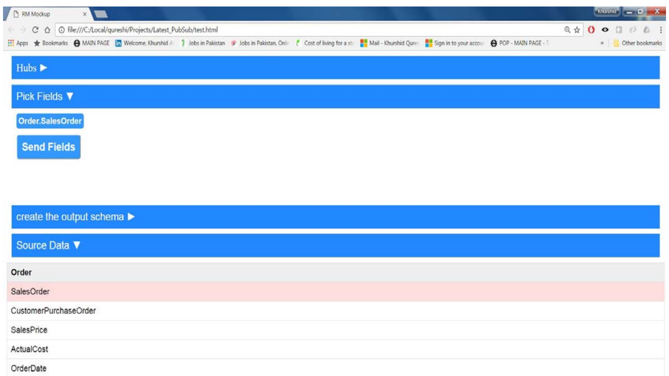

Figure 9. HTML view for selecting fields for REST Adapter

User can select the fields from the *Source Data* tab. The selected fields appear in the *Pick Fields* tab as a selection. The user can then use the *Send Fields* button to send the selection from the RM to the LSH. The LSH will fetch the data for that particular table and parse it to return the data of the selected fields back to the RM. The final result in represented in Figure 10.

```
[
    {
        "salesPrice": "100"
    },
    {
        "salesPrice": "220"
    },
    {
        "salesPrice": "540"
    }
]
```

Figure 10. Final output array of REST Adapter

V. Conclusion

The research done in this paper has given a direction of alleviating the perils involved in data collection from legacy system. It builds on an approach given by C2NET project of using cloud based platform for efficient and timely functioning of supply chain.

The paper describes the DCF of C2NET, which provides a holistic overview of the different phases of the supply chain from production to delivery. It works on the principle of cloud based information collection from heterogeneous legacy systems. In doing so, the support is provided by other major components of C2NET DCF, which include PubSub and RM.

In this research, the data collection from ERP system of SMEs is retrieved by function block approach. The implementation of two use cases, i.e., the MS Excel Adapter and REST Adapter, enable a comprehensive collection of data, which is user friendly and reliable.

In summation, this research work has relied on the approach provided by C2NET DCF for flexible and error free data collection in the supply chain. It has given way to further research arenas on C2NET implementation and has contributed in validating the project's scope. Among different features, security will be addressed in further versions in order to ensure that the platform and, hence, the interactions of different components is not vulnerable to malicious attacks. As well, the presented approach features the generality of the variations of the data resources.

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] F. Fui-Hoon Nah, J. Lee-Shang Lau, and J. Kuang, "Critical factors for successful implementation of enterprise systems," *Bus. Process Manag. J.*, vol. 7, no. 3, pp. 285–296, Aug. 2001.

[2] H. M. Beheshti, "What managers should know about ERP/ERP II," *Manag. Res. News*, vol. 29, no. 4, pp. 184–193, Apr. 2006.

[3] C. Holland, B. A. Light, and P. Kawalek, "Beyond enterprise resource planning projects: innovative strategies for competitive advantage," presented at the Proceedings of the 7th International Conference on Information Systems, Copenhagen, Denmark, 1999.

[4] J. Bisbal, D. Lawless, B. Wu, and J. Grimson, "Legacy information systems: issues and directions," *IEEE Softw.*, vol. 16, no. 5, pp. 103–111, Sep. 1999.

[5] N. Veerman, "Revitalizing modifiability of legacy assets," in *Seventh European Conference onSoftware Maintenance and Reengineering, 2003. Proceedings.*, 2003, pp. 19–29.

[6] W. Ulrich, "A status on OMG architecture-driven modernization task force," in *Proceedings EDOC Workshop on Model-Driven Evolution of Legacy Systems (MELS). IEEE Computer Society Digital Library*, 2004.

[7] 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.

[8] C. Zhao, Q. Li, M. Wang, Y. Wang, and Y. Li, "An Agent Based Wrapper Mechanism Used in System Integration," in *2008 IEEE International Conference on e-Business Engineering*, 2008, pp. 637–640.

[9] R. C. seacord, D. Plakosh, and G. A. Lewis, *Modernizing Legacy Systems: Software Technologies, Engineering Process and Business Practices*. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc., 2003.

[10] H. Guo, G. Lu, Y. Wang, H. Li, and X. Chen, "RBAC-Based Access Control Integration Framework for Legacy System," in *Web Information Systems and Mining*, 2010, pp. 194–201.

[11] Gérald Santucci, Cristina Martinez, and Diana Vlad-Câlcic, "The Sensing Enterprise," Aalborg, Denmark, 2012.

[12] *Big Dava Value Strategic Research and Innovation Agenda*, 2.0. .

[13] 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*, 2015, pp. 29–40.

[14] "D3.3 - PLANTCockpit White Paper." Oct-2015.