www.h2020first.eu

FIRST – Project Number: 734599

H2020-MSC-RISE-2016 Ref. 6742023

Manufacturing Asset/Service Description Languages

With contributions from:
John Kasse, Lai Xu, Paul de Vrieze, University of Bournemouth (BU), UK, Nicola Bicocchi, Giacomo Cabri, Federica Mandreoli, Universita di Modena e Reggio Emilia, Italy, Francesco Leotta, Mahmoud Sharf, Massimo Mecella, Sapienza Universita di Roma, Italy, Brian Setz, Michel Medema, Laura Fiorini, Ang Sha and Alexander Lazovik, University of Groningen.

Revision History	30/12/2018 First draft of ‘Manufacturing Asset/Service Description Languages’

This project has received funding from the European Union’s Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie grant agreement No 734599
H2020-MSC-RISE-2016 Ref. 6742023

Table of Contents

Introduction

Production processes are nowadays fragmented across different companies and organized in global multi-tier supply chains. This is the result of a first wave of globalization that, among the various factors, was enabled by the diffusion of Internet-based Information and Communication Technologies (ICTs) at the beginning of the years 2000. The recent wave of new technologies possibly leading to the fourth industrial revolution – the so called *Industry 4.0* – is further multiplying opportunities. Accessing global customers opens up great opportunities for firms, including small and medium enterprises (SMEs), but it requires the ability to adapt to different requirements and conditions, volatile demand patterns and fast changing technologies. Supply chains are required to be more and more *agile*, where agility is defined as a combination of responsiveness and resilience. More specifically, *responsiveness* concerns the ability to adapt to changes in the demand, provide customers with personalized products (mass customization), quickly exploit temporary or permanent advantages and keep their competitive edge, while *resilience* concerns the ability to react to disruptions along the supply chain. The resulting agile supply chains will be able to successfully adapt to an evolving and uncertain business context interms of both of demand (customization, variability, unpredictability) and supply (new components, uncertainty in the supplies, bottlenecks and risks) taking into account not only the single organization but the entire value chain.

Digital factory is a key paradigm to this end, as it aims at using digital technologies to promote the integration of product design processes, manufacturing processes, and general collaborative business processes across factories and enterprises [1, 2]. An important aspect of this integration is to ensure *interoperability* between machines, products, processes, products and services, as well as any descriptions of those. Accordingly, a digital factory consists of a multi-layered integration of the information related to various activities along the factory and related resources. At the same time, leading institutions and firms in Europe and specifically in Germany have developed and published the Reference Architecture Model Industry 4.0 (RAMI 4.0) [3]. It describes the fundamental aspects of Industry 4.0 and aims to achieve a common understanding of what standards and use cases are required for Industry 4.0. Both the technological principles of digital factories and the RAMI 4.0 architectural principles are of particular importance for our purposes. However, there are still open challenges to be addressed in order to meet the requirements of agile supply chains.

As pointed out in [4], pre-requisites for digital platforms to thrive in a manufacturing environment include the need for agreements on industrial communication interfaces and protocols, common data models and semantic interoperability. Current automated production plants, in fact, routinely employ thousands of devices from hundreds of vendors [5]. Furthermore, the growing importance of cooperation among organizations, encourages to dynamically establish inter-organisational collaborations. In this situation, interoperability becomes a relevant challenge. Service Oriented Architectures (SOAs), Internet-of-Things (IoT) technologies, and open standards for device classification and discovery have been introduced to mitigate these issues [6, 7]. The most prominent examples of these trends are described in the following, whereas a detailed survey of the field is presented in [8]. Overall, despite the recent efforts aimed at the digitalisation of manufacturing, current approaches are still lacking in one or more of the following dimensions: *(i)* they still do not pursues a seamless integrated approach starting from processes and arriving at data; *(ii)* they do not keep humans in-the-loop of product lifecycle management; *(iii)* they do not support in-process dynamic orchestration of services and data; *(iv)* they do not support alternative or personalised paths towards process goals.

The deliverable is organized as follows. First, we overview different existing approaches for asset description, classification and discovery. We then introduce a specific use case, namely, Muffin Factory, which is based on a number of existing industry use cases. A use case is then thoroughly discussed at the several levels according to RAMI 4.0 architectural principle. We propose several extensions to this model at several levels: process, services and data. In the following section we revisit and discuss the specific requirements for manufacturing/service description language. A general framework for supporting collaborative business processes with compliance checking is proposed, a possible architecture is discussed with few concrete proposals how automated verification mechanism can be realized. The section on Digital Twins discusses a different approach for describing the assets: instead of concentrating on precise descriptions (which in practice maybe difficult and/or not cost-effective) a methodology on automatic inferring of the current state of the environment is proposed. An example of existing office building is used to demonstrate the feasibility of the approach, with three potential use cases where the proposed approach and experiments may give an immediate benefit. A more formal approach based on Distributed Constraint Optimization Problem is proposed in the following section. While the discussed approach is largely based on the discrete optimization techniques and it is considered to be computationally difficult to be applied for practical applications due to their size, we show that typical structure of problems in the domain of virtual factories would actually allow us to solve this type of problems in practice. The last two sections take into account energy, which is one of the most important aspect of modern industry. We look into two problems respectively: (i) energy-aware asset descriptions and scheduling of energy-intensive operations, and (ii) energy storage systems and its parameters. Overall, the deliverable covers several important topics: from existing industry standards (such as RAMI 4.0) for description of assets/services, with an emphasis on process-based approaches to discussions over more automated and AI-based techniques where the burden of precise and thorough asset description is partially taken care of by the supporting smart infrastructure and algorithms.

Contributions from partners

Section “A view of supporting collaborative business process compliancy constraint checking” is written by John Kasse, Lai Xu and Paul de Vrieze, Bournemouth University, UK. Section “Dynamic Digital Factoris for Agile Supply Chains” is written by Nicola Bicocchi, Giacomo Cabri, Federica Mandreoli, Universita di Modena e Reggio Emilia, Italy, Francesco Leotta, Mahmoud Sharf and Massimo Mecella, Sapienza Universita di Roma, Italy. Sections on Digital Twins, Distributed Constraint Optimization, on Energy Management System and Operation Scheduling, and on Distributed Energy Storage System are written by Brian Setz, Michel Medema, Laura Fiorini, and Alexander Lazovik, University of Groningen, Netherlands. Introduction section is a joint effort of all partners. General editing is done by Alexander Lazovik, University of Groningen, Netherlands.

References

[1] N. Chungoora, R. I. Young, G. Gunendran, C. Palmer, Z. Usman, N. A. Anjum, A.-F. Cutting-Decelle, J. A. Harding, K. Case, A model-driven ontology approach for manufacturing system interoperability and knowledge sharing, Computers in Industry 64 (4) (2013) 392–401.

[2] M. P. Papazoglou, Smart connected digital factories - unleashing the power of industry 4.0 and the industrial internet, in: Proceedings of the 8th International Conference on Cloud Computing and Services Science, CLOSER 2018, Funchal, Madeira, Portugal, March 19-21, 2018., SciTePress, 2018, pp. 260–271.

[3] F. Zezulka, P. Marcon, I. Vesely, O. Sajdl, Industry 4.0 – an introduction in the phenomenon, IFAC-PapersOnLine 49 (25) (2016) 8 – 12, 14th IFAC Conference on Programmable Devices and Embedded Systems PDES 2016.

[4] E. F. of the Future Research Association, Factories 4.0 and beyond, Tech. rep., EU (2016).
[5] M. Yamamoto, H. Sakamoto, Fdt/dtm framework for field device integration, in: 2008 SICE

Annual Conference, 2008, pp. 925–928. doi: 10.1109/SICE.2008.4654787.

[6] H. Cai, L. Da Xu, B. Xu, C. Xie, S. Qin, L. Jiang, Iot-based configurable information service platform for product lifecycle management, IEEE Transactions on Industrial Informatics 10 (2) (2014) 1558–1567.

[7] F. Tao, Y. Zuo, L. Da Xu, L. Zhang, Iot-based intelligent perception and access of manufacturing resource toward cloud manufacturing, IEEE Transactions on Industrial Informatics 10 (2) (2014) 1547–1557.

[8] Y. Lu, Industry 4.0: A survey on technologies, applications and open research issues, Journal of Industrial Information Integration 6 (2017) 1– 10. doi:https://doi.org/10.1016/j.jii.2017.04.005. URL http://www.sciencedirect.com/science/article/pii/S2452414X17300043

Dynamic Digital Factories for Agile Supply Chains

Our aim is to investigate methods and techniques for enhancing global multitier supply chains by addressing the methodological issues of how to apply digital technologies into existing supply chains and proposing a reference architecture.

Asset description, classification and discovery

Device integration makes data and functionality of devices available throughout the entire automation system in ways that support association, integration, data exchange, and possibly semantic descriptions. Currently, the most widespread and relevant technologies include Electronic Device Description Language (EDDL), Field Device Tool (FDT)/Device Type Manager (DTM) and Field Device Integration (FDI).

With FDI, a technology has been developed that combines the advantages of FDT with those of EDDL in a single, scalable solution. FDI takes account of the various tasks over the entire lifecycle for both simple and the most complex devices, including configuration, commissioning, diagnosis and calibration [4]. Globally leading control system and device manufacturers, such as ABB, Emerson, Endress+Hauser, Honeywell, Invensys, Siemens and Yokogawa, along with the major associations FDT Group, Fieldbus Foundation, HART Communication Foundation, OPC Foundation, PROFIBUS PROFINET International, are supporting the development of the FDI together.

In most applications scenarios, taxonomies are usually adopted as common ground for semantic interoperability. Classifying products and services with a common coding scheme facilitates commerce between buyers and sellers and is becoming mandatory in the new era of electronic commerce. Large companies are beginning to code purchases in order to analyse their spending. Samples of taxonomy including the description and classification of manufacturing assets and services are: eCl@ss, UNSPSC, and MSDL [5]. Nonetheless, this approach to semantic interoperability cannot be employed in the considered application scenario. Indeed, most company coding systems today have been very expensive to develop and do not rapidly adapt to context changes. The effort to implement and maintain these systems usually requires extensive utilization of resources, over an extended period of time. Additionally, maintenance is an on-going and expensive process. Another problem is that company's suppliers usually do not adhere to the coding schemes of their customers, if any.

With the increasing number of assets, service discovery becomes an integral part of digital factories. Service discovery provides a mechanism which allows automatic detection of services offered by any component in the system. The objective of a service discovery mechanism is to develop a highly dynamic infrastructure where requestors would be able to seek particular services of interest, and service providers offering those services would be able to announce and advertise their capabilities. Furthermore, service discovery should minimize manual intervention and allows the system to be self-healing by automatic detection of services which have become unavailable. Once services have been discovered, devices in the system could remotely control each other by adhering to some standard of communication. Over the past years, many organizations and major software vendors have designed and developed a large number of service discovery protocols such as: SLP, Jini, UPnP and UDDI.

The Muffin Factory application scenario

MyMuffin is a company operating within the EU producing muffins willing to expand its business by allowing clients to buy muffins online. Clients can create their own muffins by

picking pre-sets of ingredients and wait for delivery[1]. The client orders box(es) (each one containing 4 muffins) online, by choosing among different possible variants, such as: *(a)* chocolate chips vs. blueberry vs. apricot bits vs. carrot bits vs. nothing as additional ingredient; *(b)* butter cream vs. hazelnut cream vs. icing sugar vs. nothing as topping; *(c)* yogurt vs. honey vs. nothing in the dough. The client can also customize the colors of the baking paper (wrapping the single muffin) as well as the colors of the box.

The muffin factory collects orders and organizes batches of muffin doughs for production. As an example, if a client asks for 3 boxes of carrot muffins with yogurt, icing sugar on top, pink baking paper, and another client for 2 boxes of carrot muffin with yogurt, nothing on top, yellow baking paper, the same dough can be used for both. Clearly this scheduling service is based on the number of (and capacity of each) dough mixers, the stream of received orders, etc. The factory has a pool of dough mixers, of different capacity. The fact that the number of different combinations is finite guarantees that such a scheduling can be performed.

When an order is received, in parallel to the dough preparation, the baking paper should be set-up as well. In addition to prepare a set of the requested baking paper, a QR-code should be printed on the baking paper and used as a unique identifier of the specific order. The identification of the single muffin is crucial for customization. After the dough has been prepared, the muffins are placed in the baking paper and sent to the oven (connected to a QR code reader) for cooking. Muffins are cooked in batches of about 1000 items and the length of this step is equal for all of them.

After the baking has been performed, the cart is operated in order to route the different muffins to the right boxes, after putting the right topping, and then to the proper delivery station. Depending on the order, different delivery agents can be used. Notably, agility is needed all along the process, e.g., the baking step may overcook some muffins, which therefore are not ready for the delivery and should be prepared again. This imply a communication with the delivery agent in order to skip the planned shipping and to set-up a new one and also a re-scheduling of the mixers in order to re-introduce the preparation of the damaged dough.

[1] MyMuffin is a fantasy company, but there are real successful examples of mass customization applied to food, cf. Mymuesli, a German company - https://en.wikipedia.org/wiki/ Mymuesli. MyMuffin is an example of a small factory in which digital transformation can be applied in order to deeply modify production processes and business opportunities. Our work can be applied to such small factories as well as more complex ones, as in the automotive industry.

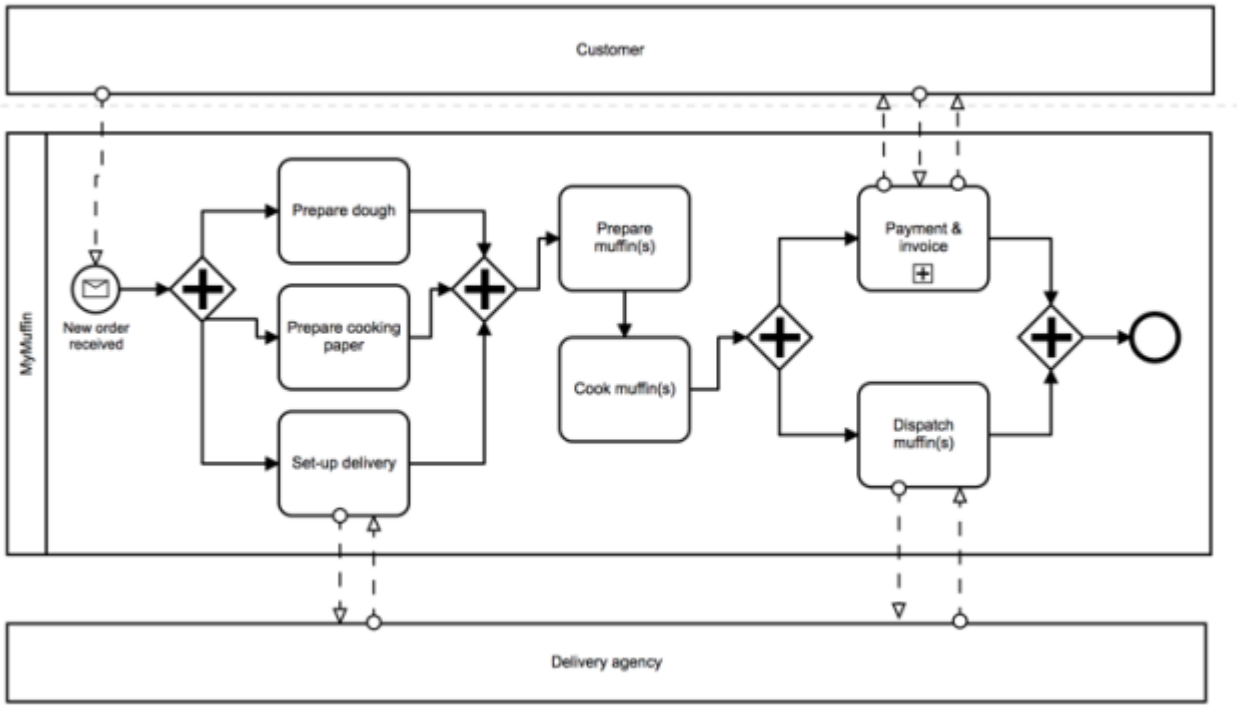


Figure above shows the process represented in Business Process Model and Notation (BPMN), cf. http://www.bpmn.org/. BPMN is a standard for business process modeling that provides a graphical notation for specifying business processes in a business process diagram (BPD), based on a flowcharting technique. A diagram is constructed with a limited set of graphical elements explained below.

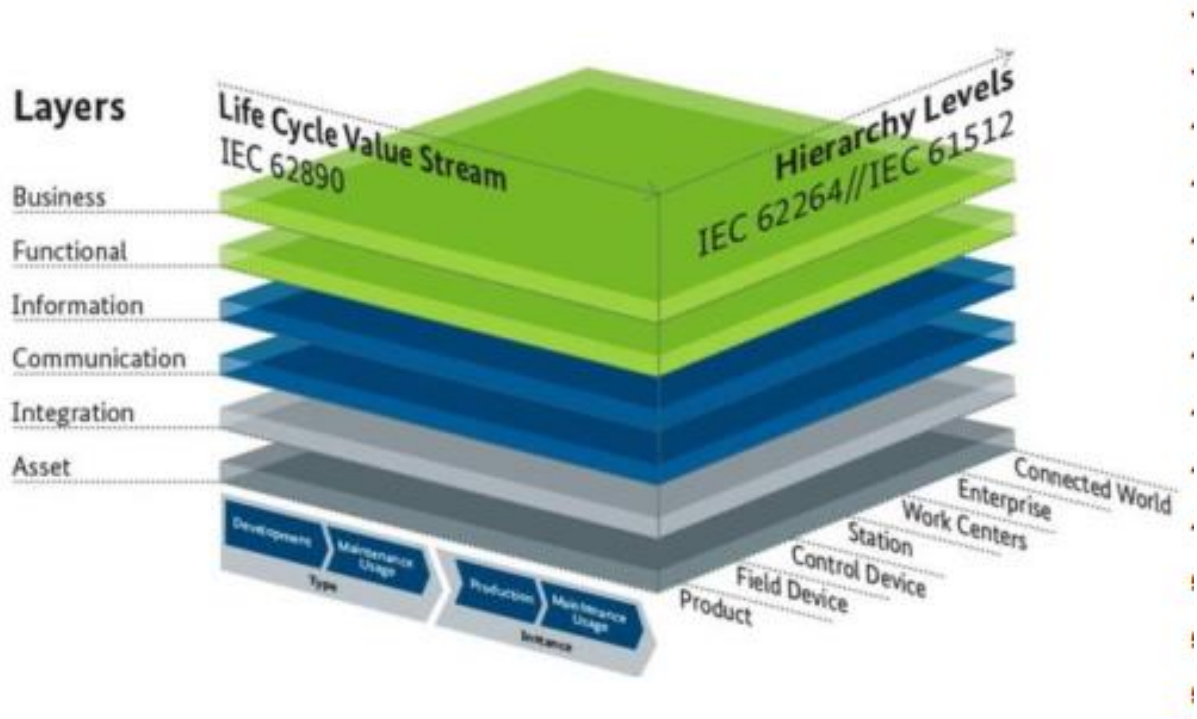
- Events, represented with circles, denote something that happens (compared with an activity, which is something that is done). Icons within the circle denote the type of event (e.g., an envelope representing a message, or a clock representing time). In the example, the start event of the process is when there is a *New order received*, and the process terminates when the flow reaches the bold-border circle.
- Activities, depicted as rounded rectangles, represent the single units of work. In our case study, they are *Prepare dough*, *Prepare cooking paper*, *Set-up delivery*, *Prepare muffin(s)*, *Cook muffin(s)*, *Dispatch muffin(s)* and *Payment & invoice*. Notably *Payment & invoice* is a sub-process, indicated by a plus sign against the bottom line of the rectangle, as it represents a compound activity, to be possibly detailed in its own diagram.
- Gateways, depicted with diamond shapes, determine forking and merging of paths. Exclusive gateways (showing an X inside the diamon) are used to create alternative flows in a process, as only one of the paths can be taken; parallel gateways (showing a + inside the diamond) are used to create parallel paths without evaluating any conditions. In the example, only parallel gateways are used, to mean that *Prepare dough*, *Prepare cooking paper* and *Set-up delivery* are all performed in parallel, then the flow is synchronized, and after some more activities performed sequentially, again *Dispatch muffin(s)* and *Payment & invoice* are performed in parallel.
- Connections are used to connect activities/events and gateways. *(i)* A sequence flow is represented with a solid line and arrowhead, and it shows in which order the activities are performed. As an example, *Prepare muffin(s)* is sequentially followed by *Cook muffin(s)*. *(ii)* A message flow is represented with a dashed line, an open circle at the start, and an

open arrowhead at the end. It tells us what messages flow across organizational boundaries (i.e., between pools – see further). A message flow can never be used to connect activities or events within the same pool. In the example, *Customer* send a message to *MyMuffin* to start the process, messages are exchanged as well during the sub-process *Payment & invoice*. Analogously, messages are exchanged between *MyMuffin* and the *Delivery agency* during the activities *Set-up delivery* and *Dispatch muffin(s)*.

- Pools and lanes are used to represent participants in a process. In particular, each separate organization is represented as a pool (rectangle), as *Customer*, *MyMuffin* and *Delivery agency* in the example. A pool can contain one or more lanes, when the designer/modeler may want to organise and categorise activities according to a function or role within the same organization. A pool can be open (i.e., showing internal details, as *MyMuffin* in the example) when it is depicted as a large rectangle showing one or more lanes, or collapsed (i.e., hiding internal details, as *Customer* and *Delivery agency* in the example) when it is depicted as an empty rectangle stretching the width or height of the diagram. Notably, no specific functions/roles are depicted for *MyMuffin*, so no lanes are represented. When an organization is depicted as a collapsed pool, it is said to offer a *public view* of its processes, to mean that no internal details are exposed. In the example, *MyMuffin*, which is the subject of investigation, is completely modeled, whereas only the public views of *Customer* and *Delivery agency* are represented (i.e., their presence and the exchanged messages).

Process specification for virtual factories

The approach undertaken in this work is based on RAMI 4.0. RAMI is a three dimensional reference architectural framework in the manufacturing industry domain developed in Germany by leveraging EU initiatives and guidelines:



We leverage RAMI 4.0 as the reference architectural framework describing how to apply digitalization technologies into existing supply chains to make them agile. According to RAMI 4.0, data is the bridge towards digitalization and is described in the integration, communication and information layers. In global multi-tier supply chains, data characteristics are largeness,

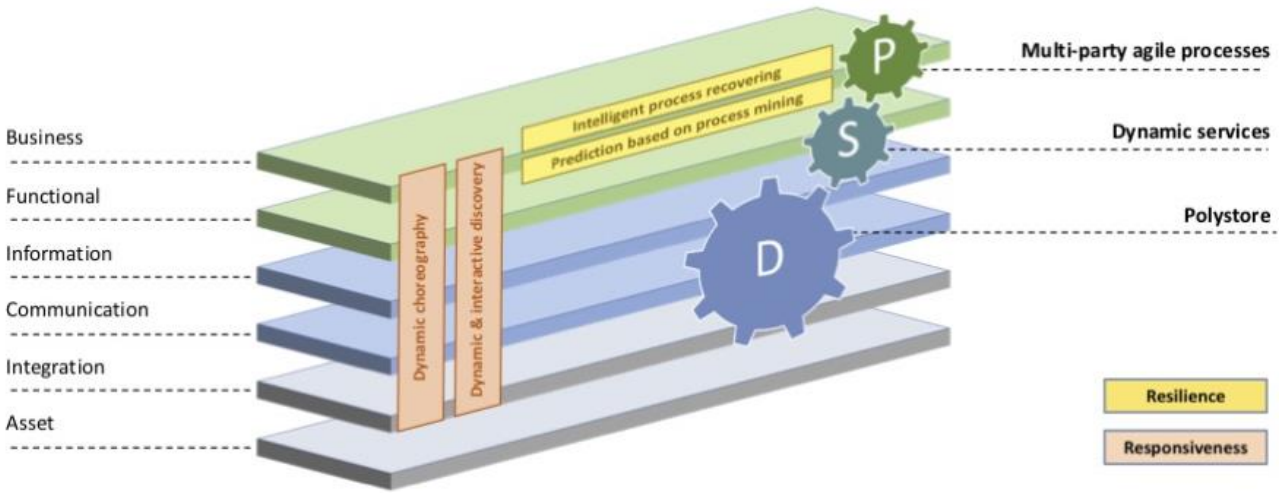
distribution and heterogeneity. For instance, machines equipped with IoT sensors continuously produce data streams, OLTP data are available in DBMSs, OLAP data are available in data warehouses, digital manuals are stored in repositories, and so on. To deal with these data features, data is organized in a dataspace of data sources that can exchange data through mappings. The dataspace adhere to the polystore architectural model supporting dynamic configurations (i.e, data sources going in and out the system).

On top, at the functional level, different kinds of services are provided to get information and to perform actions on the manufacturing parts of system (e.g., producing and assembling machines) as well as to enable interoperability with different actors of the supply chain (e.g., order management, warehouse management). Open APIs are exposed by services in order to control, discover, and compose them in a dynamic way. Rich semantic descriptions of the services should be available in order to support both the discovery of the services and their execution/invocation. This lays the foundations to achieve higher-level goals defined at business level.

At the business level, in fact, business process specifications must be able to capture not only orchestrated processes - which are bounded inside a single organization - but also choreographed processes which spans among different organizations, as a supply chain definition requires. Moreover, agility in the business processes can be achieved by shifting from the typical activity-centric process modeling to an artifact-centric modeling. This allows to model agile business processes with more emphasis to the goal to be achieved (i.e., the status to be reached). By defining several goals, with different degrees of completeness, the business process model is able to support a resilient and responsive environment as the involved parties can tune their efforts to reach one of the goals, that is not necessarily the best one. Decisions on the goal to be achieved are driven by the available data.

One of the key issues to support agile supply-chains is to provide, manage and use the different services and data that are connected to the production processes. Manufacturing machines typically provide data about their status and services. We face heterogeneous situations: from the one hand, machines are from different vendors and, even if not proprietary, they are likely to adopt different standards and vocabulary; on the other hand, services can be provided at different levels of granularity, from very fine grained one (in terms of functionalities) to very coarse. As an example, the service of the oven may expose (simple fine grained) operations for start() and stop() itself, whereas the scheduling service exposes a (complex coarse grained) operation schedule(listOfOrders): setOfMixerInstruction which takes the list of received orders and return the set of instructions to be given for the dough to the different mixers. The role of the digital factory is to integrate the different services and data and to combine them in order to make the whole process as efficient and competitive as possible in the achievement of the specific goals.
Another importation issue to be faced is the fact that the process can cover a space wider than the single factory (it supports a supply chain): usually a factory gets the raw material from suppliers and provide products or semi-finished products to customers, through delivery agents, requiring the corresponding services and data to integrate to each other or at least to be able to interact in a scalable and flexible way.

In light of the above issues, we envision a dynamic framework capable of assisting users through the discovery of service and data flows that best fit the expressed requirements and their evolution. The overall picture of the resulting RAMI 4.0-based architectural framework and the involved technological solutions are shown below. In the following the three layers are detailed.

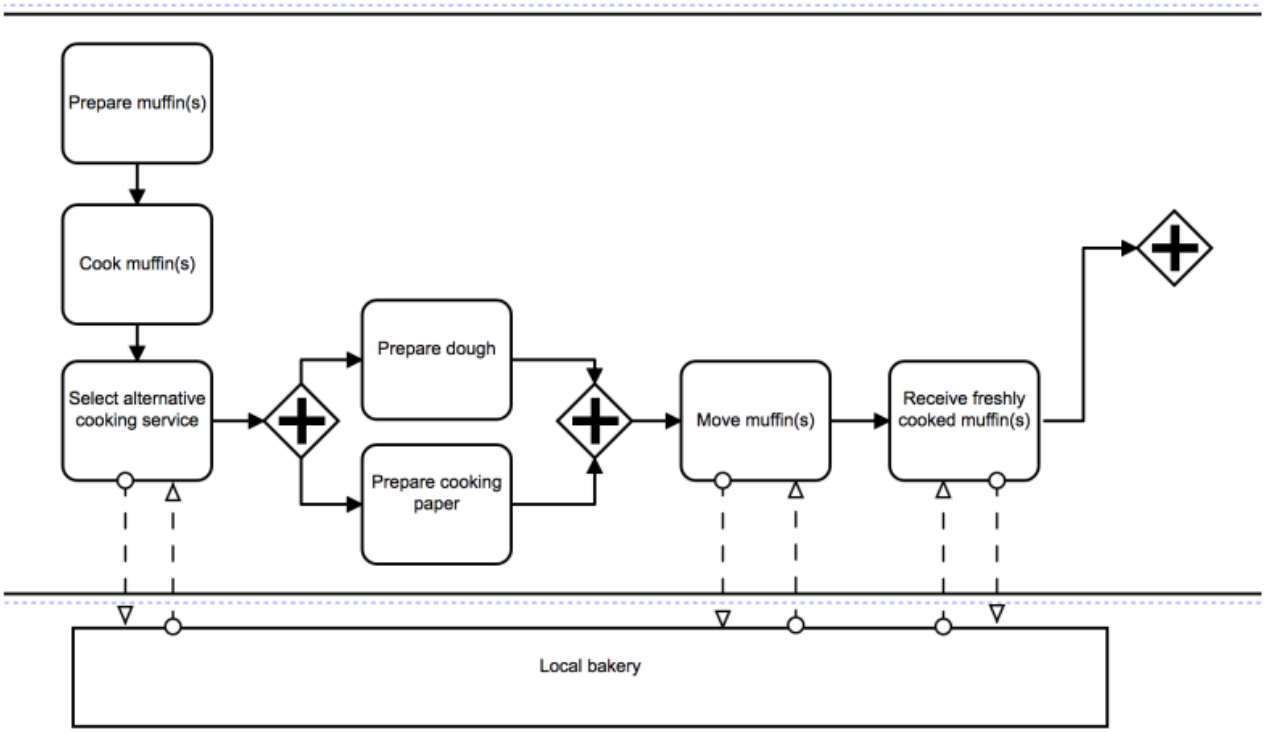


Process space layer: goal-oriented process specification

The top layer of the proposed architecture deals with the goals and the processes able to achieve such goals. In the MyMuffin example, some goals of the process are: *[G1] for each order, evade it within 36 hours (where evade means the muffins are packed and ready to be delivered)*; *[G2] for each order, the final delivery to the customer should be within 72 hours from the order.*

The MyMuffin company adopts a process in which sub-goals might have been defined for specific parts (i.e., goals can in turn be decomposed in subgoals), e.g., in order to achieve G1, it should be *[G1.1] muffins should not be overcooked*. Notably, MyMuffin would like to define, on the basis of such goals, specific KPIs – Key Performance Indicators, which qualify the QoS of the production process, e.g., the above 2 goals (i.e., G1 and G2) should be satisfied at least on 95% orders on weekly basis. Clearly, goals and KPIs are defined over many aspects, including the interactions with external companies being part of the process (e.g., the delivery agents having as goal to employ maximum 24 hours from pick-up to delivery, and to keep a KPI of 95% satisfaction over the week).

As an example of agility, we can imagine that in a given day some muffins get overcooked due to an error in the oven. This means that the goal [G1.1] is not achieved. In such a case, the digital platform will operate in order to re-arrange the process to achieve the goal. Through automated planning techniques, as the one adopted in the smartpm [2], the process can be modified as shown in below. In particular, after the original activities *Prepare muffin(s)* and *Cook muffin(s)*, new activities are introduced, in order to *Select alternative cooking service*, as a local bakery nearby MyMuffin that offers the availability of the oven; then, analogously to the original process, *Prepare dough* and *Prepare cooking paper* are performed, the muffins are moved and finally are received freshly cooked (cf. *Move muffin(s)* and *Receive freshly cooked muffin(s)*). Finally the process prosecutes as the original one. Notably, this is only one of the possible adaptations, the more complex as it re-arrange the process; in the example, it is used if simpler solutions are not possible in the given situations. We will see later that other solutions at the underlying levels are possible, depending on the specific situation.



Service space layer: service discovery and composition

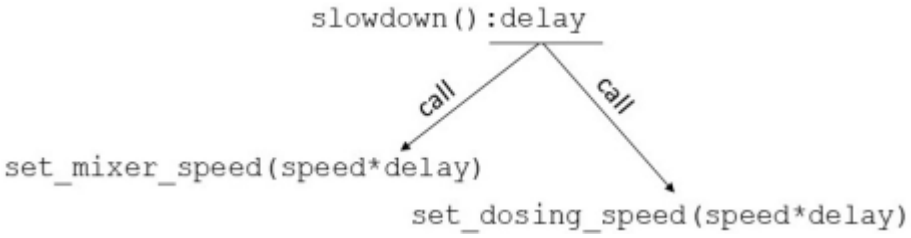
Starting from the goals and processes defined in the process layer, services must be dynamically composed to achieve goal(s). In our example, we have different machines that can expose operations such as setting/increasing/decreasing the oven temperature, starting/stopping the dough mixer and providing related data by means of OpenAPIs. Rich semantic descriptions of the services should be available, in order to support both discovery and service execution. The descriptions should include some keywords that identify the context of the service (e.g., “food”, “cooking”), the equipment (e.g., “oven”, “mixer”), the performed operation (e.g., “turn-on”, “speedup”), and the parameters (e.g., “temperature”, “speed”).

With regard to the discovery phase, the semantic description is exploited to search for specific services without knowing their exact name and their syntax a priori. Semantic techniques can be exploited to find synonyms and keywords related to the words searched for in this phase. Searches can be performed either automatically by the process layer or by human operators which may be involved when needed (e.g., the adaptation techniques realized in the process layer fail, and a human intervention is needed in order to make the process progress).

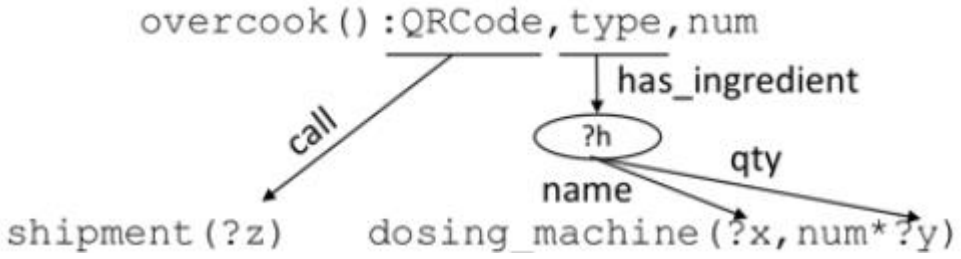
Semantic descriptions can be used in the composition phase as well. Being the composition dynamic, the platform must not only find but also use the needed service or eventually provide support to human operators. To this purpose, the semantic description of the service parameters is needed in order exploit the meta-services of the data layer to adapt the client service invocation to the server syntax (see next subsection). Some proposals and examples of semantic service descriptions have already been proposed [3].

The dynamism is useful to handle unexpected situations, often notified to a human operator. We report a couple of examples: in the former, an unexpected event causes an internal reorganization of the jobs; in the latter, an unexpected event deserves the interaction with an external actor.
The first example is related to speed adaptation due to oven performance. It may happen that the oven does not reach the required temperature because of some reasons (for instance, a cold winter day, bad isolation, broken door, and so on). The system provides the service

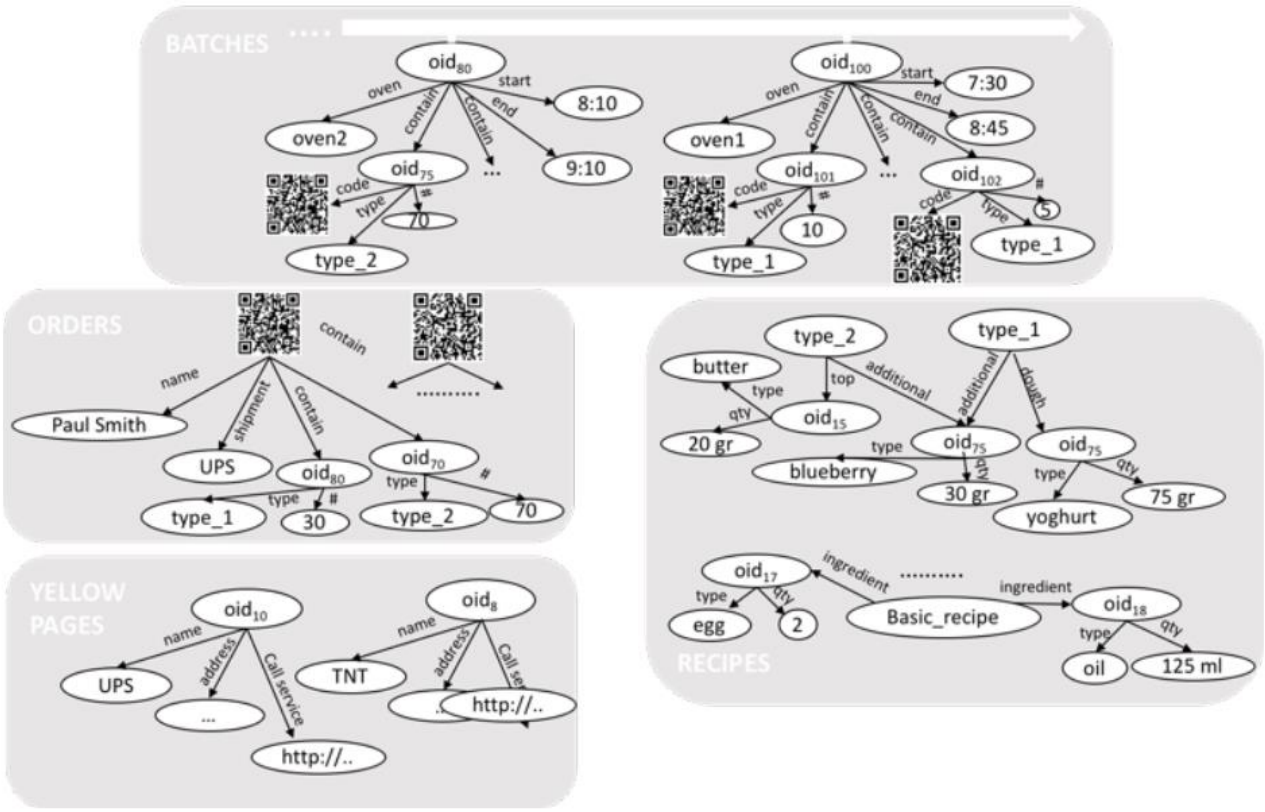
slowdown():delay, which outputs the delay in percentage; for instance, if the oven was expected to reach the correct temperature in 30 minutes, but it actually needs 45 minutes, a delay of 33% is notified. The slowdown() service is then composed with all the services available for reducing the speed of the machines; for instance, in Figure below the services set mixer speed() and set dosing speed() are invoked to reduce the speed of the dough mixer and of the dosing machine.



The second example is more complex, even if related to a simple unexpected event: some muffins are overcooked muffins, a case in which the courier must be notified to modify the shipment and a new set of muffins must be produced starting from the list of needed ingredients. To this purpose, the service overcook():QRCode,type,num is available in the platform and can be activated either by a monitoring facility or by human intervention. This service outputs the type (type) and number (num) of the overcooked muffins and the corresponding order (QRCode) and must be composed with two discovered services: one interacting with the courier (e.g., shipment(URL) with the courier Web service as input) and one activating the dosing machine (e.g., dosing machine(ingredient,quantity) with ingredient and quantity as input). The composition (see below) requires the connection of the output with the input. Essentially, the composition connects the discovered services by making explicit the relationships between the involved service parameters. ?x, ?y, ?z, ?h are variables and the corresponding values must be discovered in the data space as they represent the input to the two services, shipment and dosing machine.



Clearly, the platform must also consider failure situations, such as oven out of work, refrigerator service not found, and so on. These issues require the frequent involvement of humans in the loop in order to deal with them in an effective way, or to revert to upper layers (as shown above in the case of complete process re-arrangement).

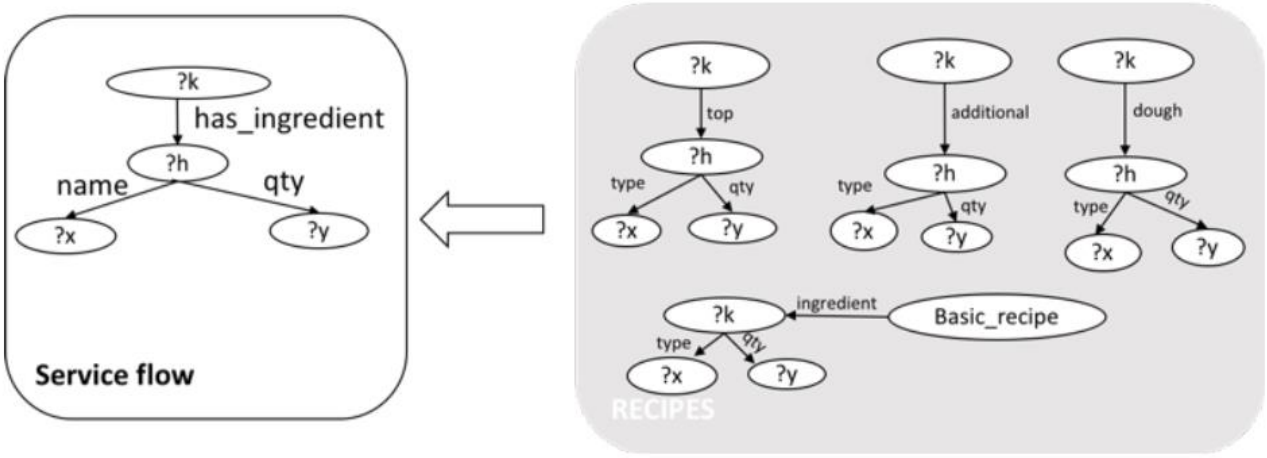


Data space layer: service-oriented mapping discovery and dynamic dataspace alignment

Data are managed and accessed in a data space. The data space must be able to deal with a huge volume of heterogeneous data by autonomous sources and support the different information access needs of the service level. In particular, a large variety of data types should be managed at the dataspace level. Data can be static such as data available in traditional DBMSs but also highly dynamic like sensor data that are continuously generated. Moreover, the dataspace should accommodate data with different degrees of structure, from tabular to fully textual data. Finally, the dataspace should cope with the very diversified data access modalities sources offer, from low level streaming access to high level data analytics.

To this extent, the data modelling abstraction we adopt to represent the data space is fully decentralized, thereby bridging, on the one hand, existing dataspace models that usually rely on a single mediated view [6] and, on the other hand, P2P approaches for data sharing [7]. The dataspace is therefore a collection of heterogeneous data sources that can be involved in the processes, both in-factory and out-factory. Those data are either describing the manufactured products or the manufacturing processes and assets (material, machine, enterprises, value networks and factory workers) [1]. Each data source has its data access model that describes the kind of managed data, e.g., streaming data vs. static data, and the supported operators. As an example, sensed parameters such as temperature in the oven, temperature in the packing station, levels of the different ingredients, etc. are all streaming data needed in the dataspace of MyMuffin that can be accessed only through simple windowing operators on the latest values. On the other hand, supplier data can be recorded in a DBMS that offers a rich access model both for On Line Transaction Process (OLTP) operations and On Line Analytical Process (OLAP)

operations.



Data representation relies on the graph modelling abstraction. This model is usually adopted to represent information in rich contexts. It employs nodes and labelled edges to represent real world entities, attribute values and relationships among entities. “Batches” is a data stream that reports the cooking status over time; “Orders” is the set of records storing the back orders made by client online and the corresponding QR-codes; “Recipes” is a semi-structured data set recording the recipes of the different kinds of muffins; “Yellow pages” is a web-based data source about the couriers and the related Web services. The oid’s, like *oid*101, are object identifies and are used to collect together data referring to the same real-world entity. It is worth noting that graph data can be serialized in a triple base where each triple has the form (*s*,*p*,*o*), where *s* is the source, *p* is the property, and *o* is the object.

The main problem the platform must cope with when dealing with data is data heterogeneity. Indeed, the various services gather data, information and knowledge from sources distributed over different stakeholders and external sources, e.g., the delivery agents and the Web. All these sources are independent, and we argue that a-priori agreements among the distributed sources on data representation and terminology is unlikely in large digital supply chains over several digital factories.

Data heterogeneity can concern different aspects: *(1)* different data sources can represent the same domain using different data structures; *(2)* different data sources can represent the same real-world entity through different data values; *(3*) different sources can provide conflicting data. The first issue is known as schema heterogeneity and is usually dealt with through the introduction of mappings. Mappings are declarative specifications describing the relationship between a target data instance and possibly more than one source data instance. The second problem is called entity resolution (a.k.a. record linkage or duplicate detection) and consists in identifying (or linking or grouping) different records referring to the same real-world entity. Finally, conflicts can arise because of incomplete data, erroneous data, and out-of-date data. Returning incorrect data in a query result can be misleading and even harmful. This challenge is usually addressed by means of data fusion techniques that are able to fuse records on the same real-world entity into a single record and resolve possible conflicts from different data sources.
For instance, if the user is interested in reconstructing the current status of back orders, then it is necessary to fuse the data stored in the batches data source and the data stored in orders data source. In this case, entity resolution is necessary because the same muffin type of the same order is represented by different oid’s (e.g., *oid*101 and *oid*80 or *oid*75 and *oid*70) and data fusion is necessary because, when the information about the same muffin type in the same order are grouped together, there will be two edge symbols, i.e., “#”, with different semantics, one representing the number of ordered pieces and the other one the number of cooked pieces.

Traditional approaches that address data heterogeneity propose to first solve schema heterogeneity by setting up a data integration application that offers a uniform interface to the set of data sources. This requires the specification of schema mappings that is a really time- and resource-consuming task entrusted to data curation specialists. This solution has been recognized as a critical bottleneck in large scale deeply heterogeneous and dynamic integration scenarios, as digital factories are. A novel approach is the one where mapping creation and refinement are interactively driven by the information access needs of service flows and the exclusive role of mappings is to contribute to execute service compositions [8]. Hence, we start from a chain of services together with their information needs expressed as inputs and outputs which we attempt to satisfy in the dataspace. We may need to discover new mappings and refine existing mappings induced by composition requirements, to expose the user to the inputs and outputs thereby discovered for their feedback and possibly continued adjustments. Therefore, the service composition induces a data space orchestration that aims at aligning the data space to the specific service goals through the interactive execution of three steps: mapping discovery and selection, service composition simulation, feedback analysis. Mappings that are the outcome of this process can be stored and reused when solving similar service composition tasks.

Essentially, the data flow indicates that from each QRCode returned by the overcook service, *(i)* it should be derived the Web service to interact with the delivery agent/courier, whereas *(ii)* from the type of the overcooked muffin it should be derived the list of ingredients together with the required quantity as input to the dosing machine.

References

[1] E. F. of the Future Research Association, Factories 4.0 and beyond, Tech. rep., EU (2016).

[2] A. Marrella, M. Mecella, S. Sardina, Supporting adaptiveness of cyber- physical processes through action-based formalisms, AI Commun. 31 (1) (2018) 47–74. doi:10.3233/AIC-170748.

[3] G. Castelli, M. Mamei, A. Rosi, F. Zambonelli, Engineering pervasive service ecosystems: The sapere approach, ACM Trans. Auton. Adapt. Syst. 10 (1) (2015) 1:1–1:27. doi:10.1145/2700321. URL http://doi.acm.org/10.1145/2700321

[4] M. Gunzert, Compatibility and interoperability in field device integration; a view on eddl, fdt and fdi, in: 2015 54th Annual Conference of the Society of Instrument and Control Engineers of Japan (SICE), 2015, pp. 941–946. doi:10.1109/SICE.2015.7285561.

[5] M. Hepp, J. Leukel, V. Schmitz, A quantitative analysis of ecl@ss, unspsc, eotd, and rntd content, coverage, and maintenance, in: IEEE International Conference on e-Business Engineering (ICEBE'05), 2005, pp. 572–581. doi:10.1109/ICEBE.2005.15.

[6] A. Marrella, M. Mecella, S. Sardina, Intelligent process adaptation in the smartpm system, ACM Transactions on Intelligent Systems and Technology 8 (2) (2016) 1–43.

[7]W. Penzo, S. Lodi, F. Mandreoli, R. Martoglia, S. Sassatelli, Semantic peer, here are the neighbors you want!, in: Proceedings of the 11th international conference on Extending database technology: Advances in database technology, ACM, 2008, pp. 26–37.

[8] F. Mandreoli, A framework for user-driven mapping discovery in rich spaces of heterogeneous data, in: OTM Confederated International Conferences" On the Move to Meaningful Internet Systems", Springer, 2017, pp. 399–417.

A view of supporting collaborative business process compliancy constraint checking

Requirements for the Manufacturing Asset/Service Description Languages to Support Collaborative Business Process Compliancy Constraint Checking

It is practice for digital manufacturing systems to collaborate by exchanging data and knowledge as part of product lifecycle management (PLM). The manufacturing asset/service description languages represent the platforms contributed by individual firms for collaboration. Thousands of devices compose a single digital factory, yet the devices may come from one or several different manufacturers. To facilitate integration, different standards are established for device manufacturers must follow while manufacturing the devices to ensure device integration and compatibility. Xu et al. (2018) present some of the standards:

- Standard for the Exchange of Product Model Data (STEP) an industrial automation systems and integration standard for product data representation and exchange,
- Open Services for Lifecycle Collaboration (OSLC) standard for specification for integration tools.
- Reference Architecture Model for Industry 4.0 (RAMI 4.0) focusses on manufacturing process and production facilities e.g. the Functional and Information Layer. Field Device Integration (FDI) is a specification intended to overcome device incompatibilities using an FDI package in which a manufacturer defines the data, functions and user interface elements for the device. Details of these standards are part of WP1 D1.1 report.

Besides device compatibility standards, manufacturing process, a form of collaborative business processes must satisfy conformance to a set of standards, industrial practices, contractual obligations and other regulations (Kasse, Xu and de Vrieze, 2018). Examples in this case include the

- Sarbanes Oxley Act
- Base III Act
- General Data Protection Regulation

Therefore, the standards form the source of compliance constraints that the manufacturing asset/service description languages must comply with while supporting the design or description of manufacturing collaborative business processes or services.
Reference to existing software-only service description approaches such as WSDL (Chinnici *et al.*, 2007) or semantic web service languages such as WSMO (Fensel and Bussler, 2002) should also form a well-formed basis of knowledge on which to base the manufacturing language.
To facilitate collaboration, manufacturing firms share business knowledge and logic in form of process models designed with support from the virtual factory systems. The collaboration is challenged by the divergent business semantics, policies and interests from the different manufacturers let alone the different systems used to construct the process models. In a collaborative environment, firms run a single model as a reference process, but individual firms create specific process variants to suit their own needs and characteristics. Both the reference model and the variants must prove compliancy with constraints and integration requirements to achieve interoperability. The proof is achievable through formal reasoning over models, an element that must be supported by the manufacturing asset/service description languages.
Therefore, compliance verification is a key requirement for manufacturing asset/service description languages should to support model checking and reasoning about compliance requirements specified in standards, semantics and business requirements.

A Framework Design of the Manufacturing Asset/Service Description Languages for Supporting Compliancy Constraint Checking

The attributes verification of the manufacturing asset/service description languages for supporting compliancy constraint checking (Gammal 2014, Kasse et al. 2017).

- *Formality*: The language should be formal to permit application of associated automatic analysis, reasoning and verification tools and techniques to enforce compliancy checking.
- *Dynamism, Flexibility and Complexity*: Several activities compose a collaborative business process, continuous changes affect process outcome. The volatility of such processes should be verified, and the language should support integration, propagation and continuous verification.
- *Expressiveness*: This describes the degree to which a descriptive language facilitates verification of several compliance requirements and properties over the collaborative business process model. Ability of the language to support expression of compliance requirements from different domains and their verification determines its expressivity.
- *General**:*** The language should be generic to facilitate capturing of compliance requirements from the wide standards, practices and business interests. For the collaborative process, the language to permit expressions, specification and checking of from all perspectives of the business process.
- *Fault tolerance and exception handling*: services are composed semi automatically or in fully automated manner following the principles of SOA. Some compliance requirements may be less-restrictive as opposed to others which are highly restrictive. Where violations occur, the language should permit fault tolerance or exception handling through which the violations can be corrected, or which part of compliance rule can be overridden.
- *Useful feedback*. When violations are detected, feedback should be provided to the user in a friendly, meaningful and understandable form.
- *Consistency checks*: Contradictions and conflicts might arise between compliance requirements particularly when they originate from different sources. It is desirable for the language to provide supporting mechanisms to identify and resolve such inconsistencies.
- *Annotation ability*: the feature that supports annotation of collaborative models with compliance requirements at the different levels of the process is lacking. Besides aiding design, annotated models are easy to understand for the non-technical end users and bridge the gap between experts and non-experts.
- *Complexity of Computations*: Verification languages employee algorithms that support model checking. Regardless of the search methods employed (whether breadth first search or depth search), the effectiveness of the search determines computation power of the algorithm and its efficiency.

Potential Approach to Achieve the Manufacturing Asset/Service Description Languages for Supporting Compliancy Constraint Checking

To facilitate and enable the manufacturing asset/service description languages to support compliance constraints checking over business process collaborations, there is need to enhance existing languages with a set of attributes or components. Figure1 shows the constraints modeling and verification architecture, such may include not limited to the following components.

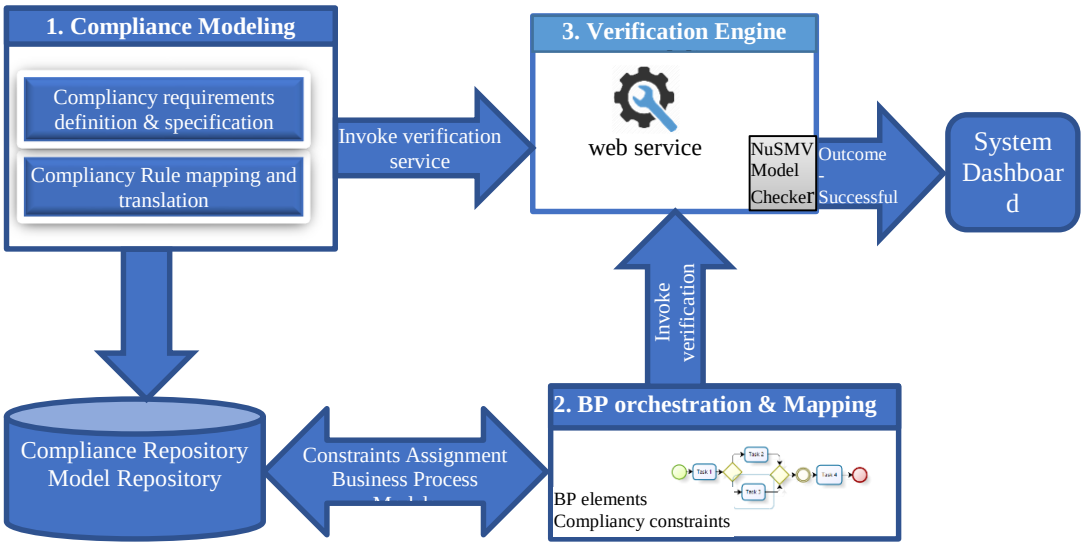


Figure 1. Constraints Modeling and Verification Architecture

Compliance repository; the repository would support creation and storage of knowledge concerning compliancy patterns. The patterns in this case are the recurring principles, articles or policies from the existing laws, standards and practices. The patterns are further broken down into compliancy attributes fitting specific business process facets of control flow, resource, data and temporal constraints. Examples of attributes are such as the ones proposed in D1.2.
Business and model repository: this repository supports the creation and storage of business knowledge covering all business interests of the collaborators expressible in form of models, choreographies, reference process models, process model variants and model versions.
Compliance verification module; The module is to support compliance constraints definitions, specification and reasoning over process models. By applying a set of algorithms, the process model or its variants are checked for errors while providing useful and meaningful feedback to support error correction at design time. The verification takes in input constraints requirements and models from the compliance repository and model repositories and returns compliancy outcome i.e. compliancy or violation.
Report generator: The purpose is to generate different forms of detailed reports as required by the end users depicting compliancy verification outcomes. Where compliance is violated, diagnosis to the source of violation should be provided in a user friendly meaningful form. It also serves as a notification dashboard for sending notifications to the user about potential flaws and violations during design.

Potential Approaches Achieving the Verification Implementation

- Adopt a constraint modelling approach (Pesic, 2008); this is a flexible approach where process models are designed following constraints requirements in a declarative manner as opposed to the imperative approach. Through constraints modelling, it is possible to achieve compliancy by design (Sadiq and Governatori, 2010).
- Extension of existing languages; service description languages lack components for model annotation and verification. Other than reinventing the wheel, extension of existing languages or integration of two or more languages for a complete working solution is probable approach to adopt.

References

Chinnici, R. *et al.* (2007) *Web Services Description Language (WSDL) Version 2.0 Part 1: Core Language*. Available at: https://www.w3.org/TR/wsdl/ (Accessed: 19 December 2018).

Fensel, D. and Bussler, C. (2002) 'The Web Service Modeling Framework WSMF', *Electronic Commerce Research and Applications*. Elsevier, 1(2), pp. 113–137. doi: 10.1016/S1567-4223(02)00015-7.

Kasse, J. P., Xu, L. and de Vrieze, P. T. (2018) *The need for Compliance Verification in Collaborative Business Processes*.

Pesic, M. (2008) *Constraint-based workflow management systems: shifting control to users*. doi: Urn:nbn:nl:ui:25-638413.

Sadiq, S. and Governatori, G. (2010) 'Managing Regulatory Compliance in Business Processes', *Handbook on Business Process Management 2*, 2008, pp. 159–175. doi: 10.1007/978-3-642-01982-1_8.

Xu, L. *et al.* (2018) 'Interoperability of Virtual Factory: an Overview of Concepts and Research Challenges', *International Journal of Mechatronics and Manufacturing Systems*.

Digital Twins

A physical object can have a virtual model, known as a Digital Twin. This Digital Twin synchronizes the state of the physical object with the virtual representation of that object, using various Internet of Things (IoT) sensors and actuators. The virtual model can be used to monitor the physical object in near real-time and provides insights into the operational data of the object. Changes to the physical object can also be simulated, and the impact that such changes would have in the real world can be reviewed. This not only allows existing objects to operate more efficiently, but data from multiple different objects can be combined to create new digital objects that meet specific requirements from which a physical counterpart can then be created. Relevant experiments can be performed on a Digital Twin, and only then applied on a physical object. Moreover, a link between a physical object and its Digital Twin can be continuously maintained, making it possible to experiment and optimize the properties of a physical counterpart at any time.

Digital Twins are not limited to a single object, as complete buildings can have their virtual counterpart, which is composed of other Digital Twins representing a part of the building (e.g. an office). Applying the concept of Digital Twins to office buildings can be very beneficial as office buildings consume a large amount of energy and recent developments in the area of IoT and smart buildings have already shown that optimising such buildings can significantly reduce the total amount of energy that is being consumed. However, a certain amount of sensors and actuators have to be installed in an office building to realise this potential, which requires a substantial up-front investment and then results in high maintenance costs. In practice though, these physical counterparts, e.g., offices, are not completely independent or significantly different from each other. Often, instead of installing IoT devices in all parts of a building, the similarity of areas within a building can be exploited to infer the state of those areas that are not directly observable. If applied on a large scale, the cost of developing new Digital Twins for buildings could be greatly reduced, which, in turn, can make adoption of the concept much easier.

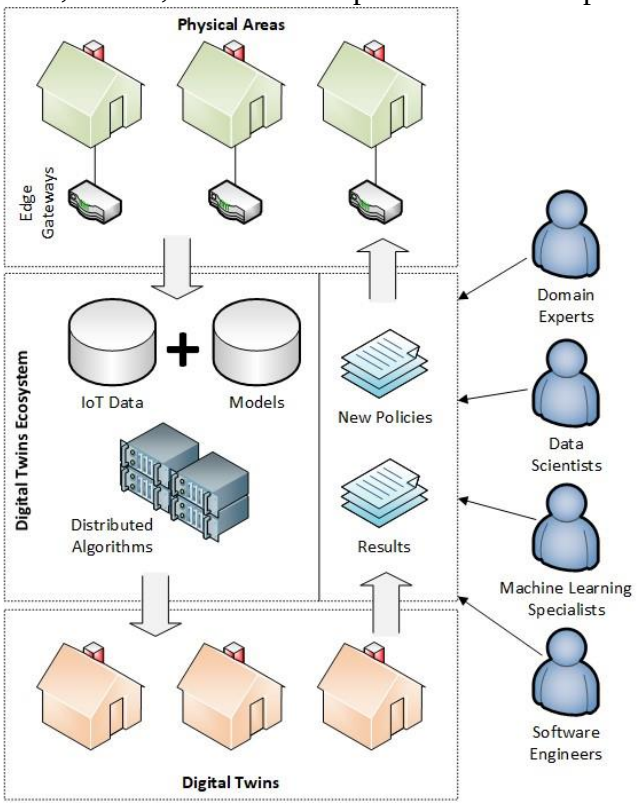


Figure 1: A global overview of the Digital Twins ecosystem.

One of the possible use cases for applying Digital Twins in practice is optimization of building maintenance costs. Changes to the building policies (e.g. for heating and ventilation) can be simulated to observe the effects on, for example, the energy consumption and the comfort of occupants before being applied to the physical building. Refined policies designed for one office can likely also be applied to other offices that are similar, thereby potentially reducing the computation cost involved in optimizing all the offices.

Successfully extracting value from the data that is available from the Digital Twins requires knowledge from many different domains: software and Big Data engineers, machine learning and data scientists, domain-specific experts (e.g., energy, smart buildings, IoT). The platform aims at having an ecosystem of Digital Twins, where models can be shared and similarities can be uncovered. Such a platform would also empower its users to perform comparisons between Digital Twins and apply changes to multiple Digital Twins in parallel.

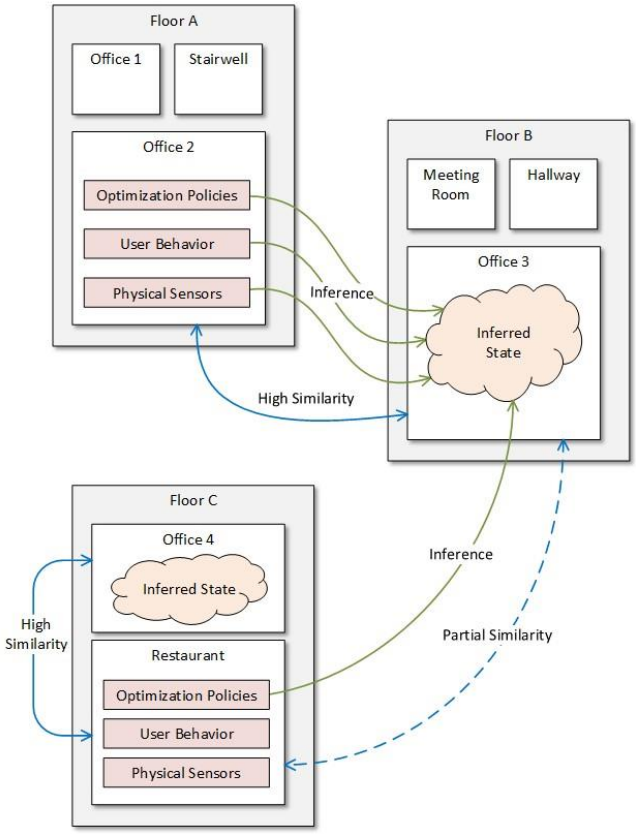


Figure 2: Inference of state by similarity.

The Digital Twins ecosystem brings together numerous distinct stakeholders that interact with different parts of the system, as is shown in Figure 1. We identify physical floors, rooms, or areas, containing sensors and/or actuators which are connected to the edge gateway. The collected data is combined with the models for the Digital Twins, and are updated using distributed algorithms. These models and algorithms are defined by domain experts, e.g., by machine learning specialists. The results of the model updates are propagated upstream, back to the edge gateways which are responsible for executing the newly generated policies in automated or semi-automated manner, e.g., if policies are accepted by the domain expert.

Comparison of different offices within a building is not limited to only basic sensor data, but also includes similarities at the level of more complex models such as physical models of the areas within the building, machine learning models, user behavior patterns, optimization policies, etc.

In Figure 2, we show a simplified example of how we can utilize the concept of similarity between rooms and areas in order to infer a previously unknown state. Floor A, Office 2 contains a number of installed sensor and actuators, and it has already calculated models that define user behavior, and has a number of optimizations policies in place. On Floor B, there is an Office 3 which has no sensors installed. Despite the lack of sensors, we can infer the state of Office 3 as it has a high similarity with Office 2. There is also a partial similarity between Office 3 and the Restaurant on Floor C. The optimization policies from this restaurant are applicable to Office 3. The quality of the resulting Digital Twin for Office 3 will depend on the degree of similarity with other Digital Twins. In many situations, even partial similarity may be sufficient to adjust the relevant properties of Office 3.

We envision a large similarity network connecting different offices based on a number of parameters - including sensor data, user behavior, energy optimization policies, etc. Similar Digital Twins are forming clusters of objects with similar properties, which can be used for inference, simulation and optimization. An example of a network of office rooms is shown in Figure 3, where clusters of different room types can be identified. These clusters have a high degree of similarity. Relationships may also exist between rooms of different types.

From the data driven machine learning perspective several techniques are potentially interesting to detect and use similarity of offices for the approximation and optimization of energy consumption in a building. In contrast to engineering modeling approaches computer science techniques aim in extraction of information based on collected measurements, focusing on the question: what can still be achieved facing imperfect data? Beyond the detection of simple correlations machine learning provides efficient algorithms to for example:

- cluster groups of offices
- detect anomalies/novelties in spatial and temporal data and
- discover feature relevance to distinguish the most informative measurements

Using such methods we can investigate the usefulness of very cheap and easily acquired measurements such as e.g. spatial and temporal occupation of the building by logging the connection of mobile devices to wireless emitters, which are standard in office buildings nowadays. In combination with publicly available weather data, forecasts and individual building specifications we, for example, can estimate the necessary amount of heating to provide comfortable working environments based on the behavior of the inhabitants.

The initial experiments has already been performed to see if the similarity between offices may be used to better control heating in the building, focusing on the construction of a feature engineering framework to ease the selection of features that are relevant for a certain model. For this research, the realisation of a smart heating system in the Nieuwenhuis building of the University of Groningen (an office building) was selected as a use-case. The smart heating system requires an accurate model for the room temperature of the offices within the building to optimally control the heating system. Sensors were deployed in the offices to measure the room temperature and the room occupancy. Actuators were also installed to regulate the valves of the radiators in the offices. Based on this data, the similarity of offices, as determined by clustering, is used to find features that may be common between offices and that can have a significant influence on the temperature, such as the influence of the sun and the orientation of the office.

Figure 4 shows a map of a part of the office building enriched with a visualisation of the temperature behaviour of several offices. The colour of the outer ring of the circles represents the cluster to which the office belongs, and the inner part visualises the temperature behaviour of that

office. Based on the clustering results and the visualisations, it becomes clear that offices that have been assigned to the same cluster show similar behaviour in terms of room temperature. On average, the room temperature of the offices assigned to the red cluster is slightly higher, possibly due to the influence of the sun. Applying the clustering techniques to the entire building produces the results that are shown in Figure 5 (only the clusters are displayed). It shows that the green cluster is predominantly present on the north side of the ground floor and that the black cluster is strongly represented in a specific area on the first floor. This indicates that these offices share some common characteristics, for example in terms of room temperature behaviour.

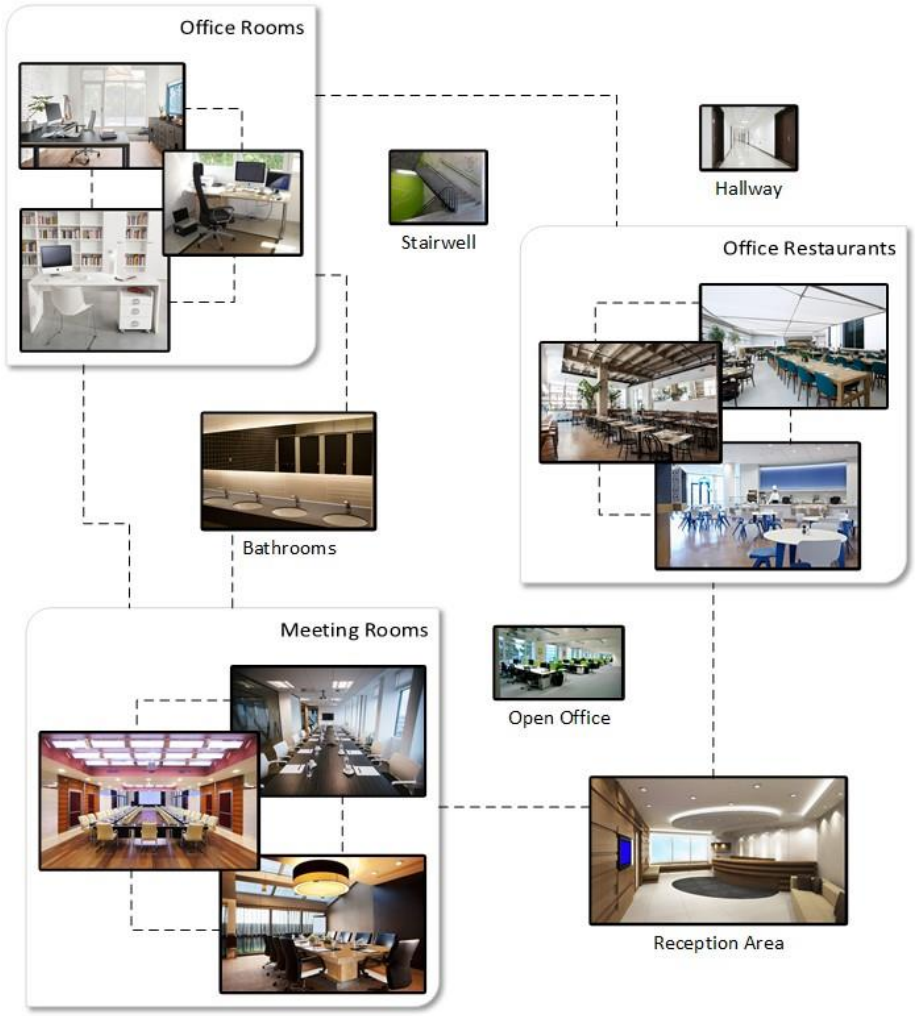


Figure 3: A network of offices based on similarity metrics.

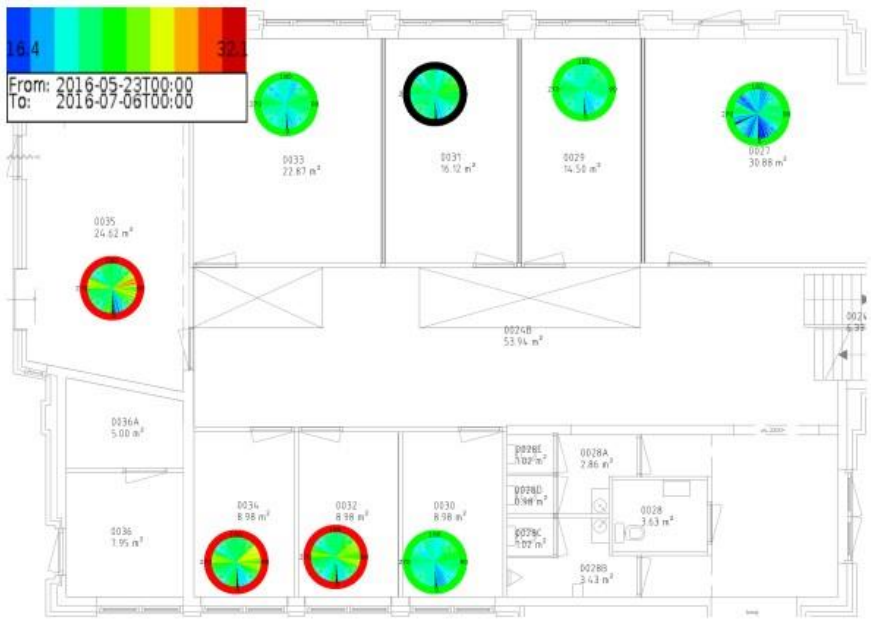


Figure 4: Similarity of offices visualised together with the room temperature behaviour.

In a different building of the University of Groningen, the Bernoulliborg, a smart lighting solution was developed. The system is deployed in the building's restaurant which covers a total area of 251.50m2 and offers a capacity of 200 seats. Numerous motion sensors and light sensors were deployed, and the existing lamps were modified to enable remote actuation. An overview of all sensors and actuators is shown in Figure 6. Orchestration of the lamps in the restaurant is performed using Hierarchical Task Network planning combined with activity recognition. As a result we were able to obtain significant savings in terms of energy consumption. We compared our approach (automated control) to the original situation before the deployment of sensor and actuators (manual control) and to a setup comprising of motion sensors exclusively. Figure 7 shows the comparison of the three approaches in terms of energy consumption. In the end, a reduction in energy consumption between 70 to 80 percent was achieved.

In the same building, potential economic and energy savings have also been investigated for several offices and a social corner assuming the availability of a smart grid. Sensors are installed in each space to detect the presence of occupants and collect data about the energy consumption of the appliances (the part of the set-up is displayed in Figure 8). Policies are also associated with each appliance (e.g. a fridge has to be operated a certain amount of time every hour to ensure proper temperature). Based on this data, the policies of the appliances and the dynamic pricing obtained from the smart grid, a scheduling algorithm computes an optimal schedule that minimises the total cost of the energy that is used. Experiments have been performed for four weeks, where the data collected during the first two weeks is used to determine policies for the appliances, which are used by the scheduler during the last two weeks to control the appliances. Economic savings of up to 50 percent are observed when the system is controlled by the scheduler. The reduction in terms of energy usage is only around 10 to 15 percent, but this is mostly due to the irregular usage of the appliances. Additional experiments were performed using this living lab to simulate the control of appliances using activity recognition and HTN planning to discover the amount of energy that can potentially be saved (e.g. a workstation that is powered on without any presence in the office indicates a potential saving). Applying the simulations to data collected during the office hours of three days indicates that the total energy consumption can be reduced by 75 percent, consisting of a 45 percent saving on the energy consumed by the workstations and 98 percent reduction of the energy used by the lights.

It is important to note though, that even if the machine learning and artificial intelligence algorithms are playing an important role, they are only one part of such a platform. Supporting a full ecosystem of Digital Twins requires an infrastructure which is scalable and distributed by design. Such an infrastructure should not only support the simulation of models but also has to handle data collection from IoT devices, facilitate the transfer and storage of collected Big Data, support real-time streaming and online learning, ensure the privacy and security of the data and its users, etc. Thus, we foresee a distributed cloudbased architecture that can scale on demand and adjusts to the requirements of the end-users.

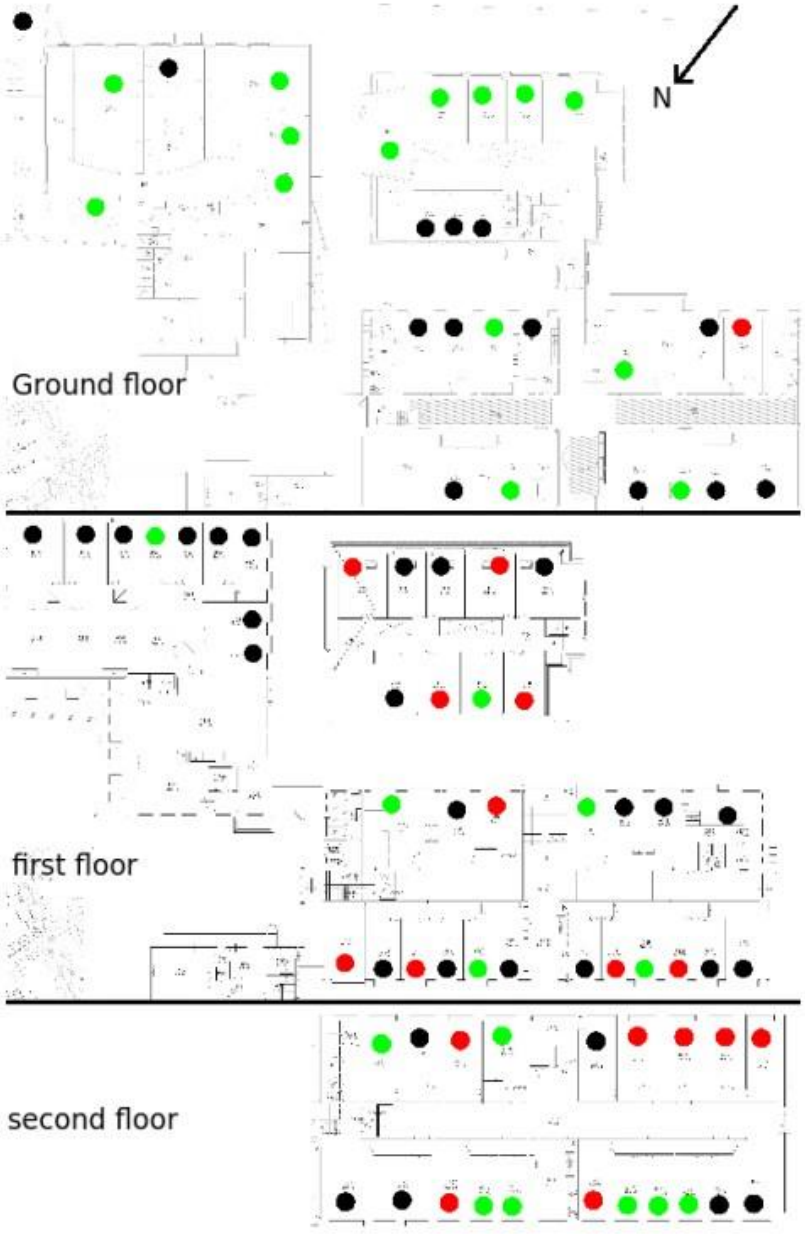


Figure 5: Results of similarity clustering of all offices within the Nieuwenhuis building.

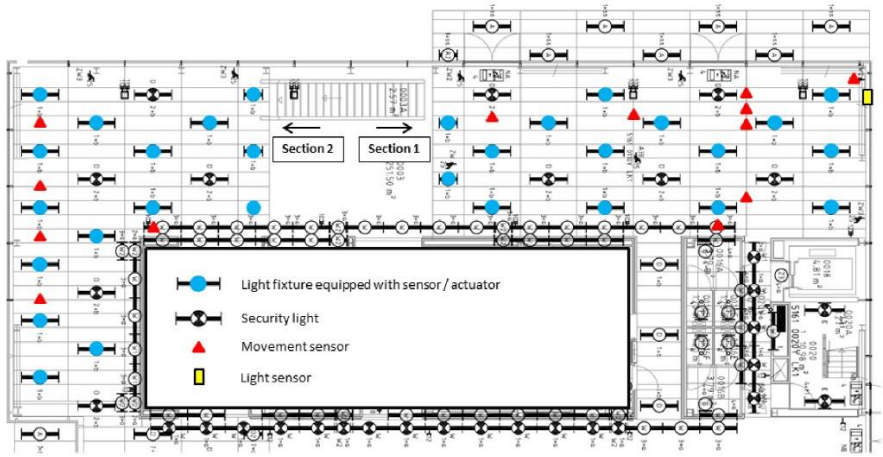


Figure 6: Deployment of sensors and actuators in the restaurant of the Bernoulliborg.

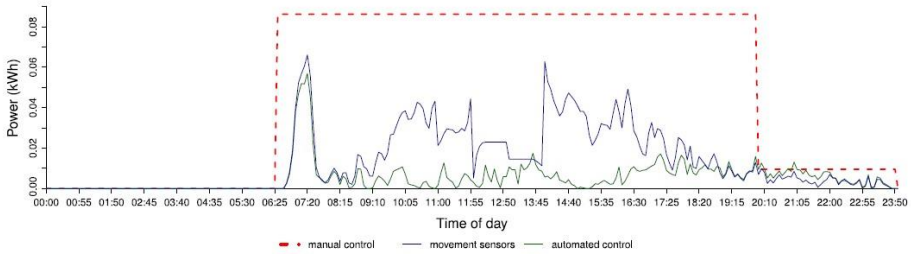


Figure 7: Comparison of energy consumption in the restaurant of the Bernoulliborg for different types of control.

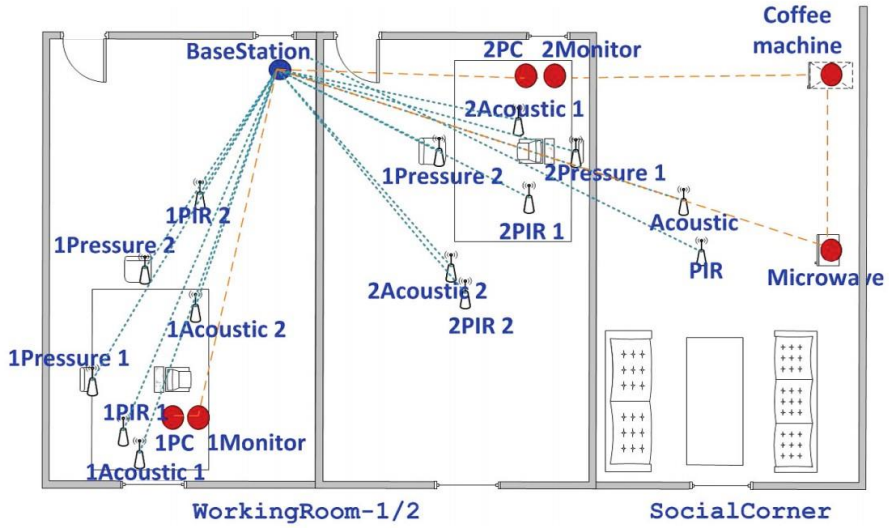


Figure 8: Schematic representation of the offices, social corner, and deployed devices.

The Digital Twins ecosystem should adhere to the principles of data locality: moving the algorithms to the data is usually much cheaper than moving the data to the algorithms. The data locality is achieved by means of edge gateways. The edge gateway is located in the building for which it is responsible. As a result, the data can be processed near the source of the data, reducing the latency, which is critical in a real-time system. Multiple edge gateways can also be utilized to distribute the algorithms, increasing the computational capacity and decreasing the time-to-solution.

Real-time data processing and online model updates require a minimal delay between the moment data is generated to the moment the newly generated data has been processed. Therefore, the concept of Edge Computing plays an important role in the Digital Twins ecosystem, as it enables the processing to take place at the edge of the network where the results are expected, on socalled Edge Gateways. This greatly reduces delay and distributes the workload across multiple Edge Gateways, enabling data processing to take place even when the connectivity to the cloud-based Digital Twins platform is lost. The Edge Gateways also enable data storage close to where the data is produced.

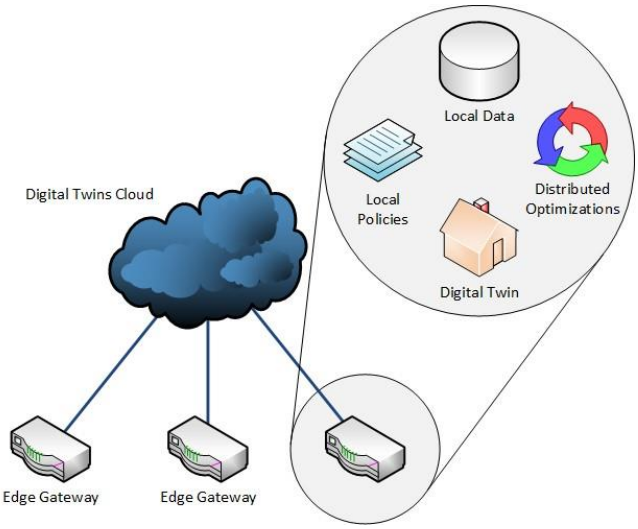


Figure 9: Edge Computing in the Digital Twins ecosystem.

Realizing such an ecosystem can be extremely beneficial for making office buildings more energy efficient. The Digital Twins allow the buildings to be monitored and changes to the policies can be simulated to review the effects without causing discomfort to the occupant. The ecosystem also allows the available data to be used by other models and algorithm, for example, for analytics or prediction purposes. Inferring the state of Digital Twins from similar ones also greatly reduces the number of sensors that have to be installed in buildings. Incorporating objects or processes other than office buildings is another possibility, allowing the ecosystem to be used for other domains as well or combined with office buildings for increased optimization.

Computation on similar or related Digital Twins can also be shared. For example, energy optimization can also be performed on a group of Digital Twins. For example, matching the supply and demand of energy across a number of offices. In that case, the optimization problem becomes much more complex as it requires the data from all the objects involved. Decomposition of the optimization problem would be required to reduce the computational complexity, and the distributed network of edge computing devices can be utilized to solve the subproblems close to the physical objects, thereby removing the overhead of collecting all the data at a single machine.

Possible Use Case 1: Similarity (heating)

The network of offices enables us to discover similarities between offices and rooms within offices. For example, there is a large company that is responsible for managing a large number of offices. There are many similarities between these offices, especially for those offices located near each other. Let us assume that our goal is to optimize the heating level in a particular office, e.g., for Office 2 from Figure 2. Let us also presume that this office does not have any sensors installed. However, we may infer the pattern of the user behavior for this office from other offices of the same company with similar employees - presuming that their day-to-day activities are similar to each other. From other offices that are built with similar materials we can infer the overall heating control policy that we can use as a basis for this office (or this part of the building). And, finally, from the light sensor measuring light intensity installed in a restaurant (which is located on the same side of the building!) we can infer whether the day is sunny or not (and if it is sunny we can reduce the amount of required heating). By combining the data collected from similar offices altogether we can calculate the optimal heating level for a particular office on its Digital Twin and then apply the resulting heating policy to a concrete physical office.

Possible Use Case 2: User-controlled complex policies

Jan is a facility manager for a large company, where he is responsible for supervising many different aspects of the company's building. One day, his reports show that a part of the building is not adhering to the sustainability goals that have been put forth by the board of the company. Apparently the offices in this part of the building have consumed significantly more energy than was anticipated. First, Jan has to determine why the energy consumption is higher than expected. He starts his investigation by looking at similar offices on the Digital Twins platform, and notices that all other offices have decreased their energy consumption over time. Next, he compares which policies have been implemented on the other offices, and discovers that there are numerous policies which have not been applied to the offices with the increased energy consumption. Jan applies the policies, which have been calculated for other offices by the machine learning specialists, to the Digital Twins of the problematic offices. He notices that some policies do not have any effect, and discards these. However, some policies are showing a great increase in energy efficiency when applied to the Digital Twins. Thus, the facility manager is able to quickly select the right policies for the particular offices in order to meet the sustainability goals.

Possible Use Case 3: Global energy optimization

Considering an optimisation problem that involves, let us say, an office building containing two hundred offices and twenty edge gateways located in different parts of the building, the goal is to optimise the energy consumption globally across all offices. Unfortunately, directly solving such an optimisation problem is computationally infeasible as it is an NP-complete problem. However, not all the offices within the building have a direct influence on each other, thus forming semi-disconnected islands that can be optimised mostly independently from each other (although sometimes they may still be somewhat connected, e.g., due to the behaviour of the occupants). The edge gateways are tasked with solving these smaller optimisation problems as these gateways are close to the required data and can adjust policies with minimal latency. Solving all of these problems may not be necessary, however, as the similarity of offices allows the policy that has been found for one office to be used for others, without the need for solving the optimisation problem again. Alternatively, if the policy is not directly applicable due to considerable differences in the state of the individual offices, a more abstract policy can be constructed that can be adjusted based on the specific state of those offices, potentially reducing the computational effort that is needed to solve the optimisation problems for similar offices significantly.

Distributed Constraint Optimization Problem

Constraint Satisfaction Problems (CSPs) are a type of search problem where, given a set of variables, respective domains of those variables and a set of constraints, the goal is to find a valid assignment for each variable such that all constraints are satisfied [1–3]. Many problems such as planning, scheduling and resource allocation can be solved by modelling it as a CSP and searching for a solution [3]. Solutions can also be differentiated with the use of an objective function, allowing optimisation of the problem by searching for the solution that minimises or maximises this objective function. Unfortunately, constraint satisfaction is an NP-complete problem, meaning that an algorithm guaranteed to find the optimal solution to a CSP requires exponential time to do so [1]. Additionally, the problem domain may be highly dynamic, making it necessary to solve the problem continuously. For certain domains, finding a solution in real-time is also an important requirement. All of this makes modelling large problems difficult and limits the applicability of CSPs.

Some problem domains have an inherently localised structure consisting of small, relatively independent regions of interrelated events that allows for a more efficient search [4,5]. The constraint graph of such a problem consists of small groups of interrelated constraints where all constraints within one group are defined on only a small subset of the variables. This has, for example, been shown to be applicable to smart environments (e.g., virtual factories, smart buildings, etc.), where processes are mostly confined to a specific area (e.g. performing an activity in one area often has a limited influence on adjacent areas) [5]. Such a domain structure allows the complete problem to be decomposed into smaller problems that can be solved mostly independently of each other, and the solutions to these sub-problems can be combined into a solution to the complete problem [5]. The dependency graph data structure captures this locality by modelling dependencies between the rules and variables of a problem domain, providing the information needed to decompose the problem (rules are more abstract than constraints and can capture additional information about the environment not included in the CSP, such as sensor information for a smart system) [5].

When the structure of a problem domain is inherently localised, the complete problem can be decomposed, and the sub-problems could be solved with any existing constraint solver. Ideally, these sub-problems are solved in parallel, however, to reduce the overall search time, especially when the complete problem is large, but distributing the work among multiple machines is non-trivial. The dependency graph data structure can also be dynamic when the interconnectivity of the rules depends on the environment, requiring the decomposition to be performed every time the connectivity changes, which adds a significant amount of overhead.

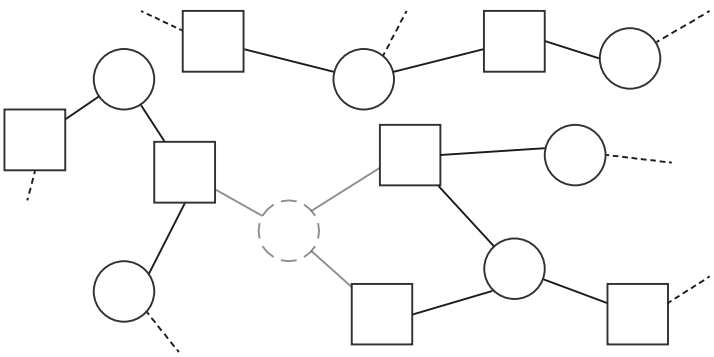


Figure 1: Example dependency graph with three disconnected components.

The dependency graph data structure captures the locality of a problem domain by modelling the dependencies between variables and rules (see [5] for a formal definition of dependency). Rules capture the constraints but also the activeness of those constraints. Inactive rules partition the dependency graph into active subgraphs. Each active subgraph represents a part of the graph consisting only of active rules that share at least one common variable. A variable belongs to the active subgraph iff a rule node of the active subgraph depends on that variable. The edges of the active subgraph contain exactly those edges connecting the rules to the variables on which those rules depend. These active subgraphs, formed by the locality and activeness of rules, produce useful structures within the dependency graph. All active subgraphs together form the set of independent components, visualised in Figure 1 (squares depict the variables and circles the rules). No dependencies exist between these components, either because no rule connects the graphs or because it is inactive, making it possible to solve them completely independent of each other. These independent components can be further subdivided into two groups.

The so-called islands are the first group, consisting of all the active subgraphs that correspond to a single local region of the problem domain (e.g. a single room or area within an office building). Vertices within these subgraphs are strongly connected and are assumed to exhibit small-world properties, meaning all vertices can be reached within a small number of steps from any of the vertices in the graph and neighbours of a vertex are likely connected to each other. The islands are the smallest problems that have to be solved and cannot be decomposed any further in a straightforward manner.

The second group, as shown in Figure 11, are the connected components, active subgraphs that consist of multiple islands that are connected to each other by several rules. It can be stated that such a graph consists of two island-like structures together with a connecting element in between. This means that the two island-like subgraphs do not overlap and are not directly connected to each other, but are connected to each other via the connecting element.

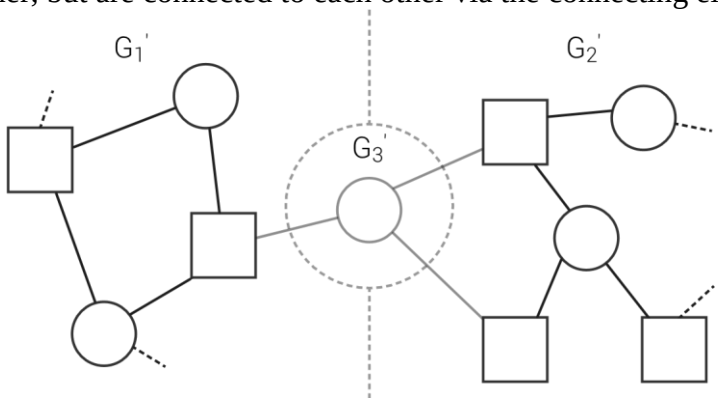


Figure 11: Connected component consisting of two islands (G'_1 andG'_2) and one rule, part ofG'_3, bridging the two islands.

Along with the previously mentioned structured, the following properties are needed to successfully reduce the size of the problem and solve large-scale CSPs (solving CSPs is possible without these properties, but decomposition becomes more difficult and the search time increases exponentially with the size of the complete problem in the worst case).

1) *The problem domain that is being modelled must be inherently localised such that interactions are bounded to small regions with only limited interactions between different regions (i.e. islandlike structures exist within the dependency graph).*

2) *The rules must have some notion of activeness to allow parts of the dependency graph to be disconnected.*

3) *Rules bridging the islands of a connected component should be mostly inactive to allow the problem to be efficiently solved most of the time. When such rules are active, the activeness should be brief and not too many islands should be connected simultaneously to prevent the construction of a large subgraph.*

In practice, the domain of virtual factories holds these 3 properties making the discrete optimization techniques (and, specifically, distributed constraint optimization) a viable approach for this domain. To be able to apply these techniques, corresponding services and assets should provide additional semantics description to incorporate according to formal definitions of [5].

References

[1] S. C. Brailsford, C. N. Potts, and B. M. Smith, "Constraint satisfaction problems: Algorithms and applications," *European Journal of Operational Research*, vol. 119, pp. 557–581, Dec. 1999.

[2] M. Yokoo and K. Hirayama, "Algorithms for Distributed Constraint Satisfaction: A Review," *Autonomous Agents and Multi-Agent Systems*, vol. 3, pp. 185–207, June 2000.

[3] V. Kumar, "Algorithms for Constraint-Satisfaction Problems: A Survey," *AI Magazine*, vol. 13, p. 32, Mar. 1992.

[4] Lansky Amy L., "Localized event-based reasoning for multiagent domain," *Computational Intelligence*, vol. 4, pp. 319–340, Apr. 2007.

[5] V. Degeler and A. Lazovik, "Dynamic Constraint Satisfaction with Space Reduction in Smart Environments," *International Journal on Artificial Intelligence Tools*, vol. 23, p. 1460027, Dec. 2014.

Energy Management System and Operation Scheduling

Climate change is one of the major challenges that the planet is facing, due to the increasing emission of green-house gases, main responsible for the greenhouse effect. Among these gases, CO_2 contributes for the largest share, and it is mainly caused by human activities [16]. The energy sector is responsible of roughly two-thirds of all GHG emissions [15]; in Europe, residential and commercial buildings cause 36% of CO_2 emissions [11]. Traditionally, heating and cooling demand have been considered the main reasons for domestic energy consumption, and, hence, for CO_2 emissions [14]. However, the increasing ownership of appliances and their use have caused a significant rise in households' need for electricity [14], whose environmental impact varies according to the production sources. When using electricity bought from the grid, the amount of CO_2 emitted to generate a kWh is determined by the current generation mix. For instance, given a certain amount of power to be produced, the emission intensity factor associated with a kWh at windy or sunny day is likely to be significantly lower than in a windless and cloudy one, when almost all the power is generated by plants burning fossil fuels [12].

Demand side management programs aim to control the residential electric use in response to price signals or incentive payments. In a price-based scheme, the consumers are offered time-varying rates to modify their demand over time, e.g., to shift the consumptions towards off-peak times; in an incentive-based one, rewards are given to those users who accept to reduce their consumptions when requested by the utility [25]. The main drivers for Demand Side Management programs are the risk of congestions along lines and the possibility to postpone expensive investments costs for the expansion of the infrastructure capacity, while giving the consumers the chance to reduce their energy bills. By an environmental point of view, while reducing the consumptions has certainly a positive effect, a price-based shifting of the demand over time might not always lead to the solution with lowest environmental impact [28]. Several works propose electric load management strategies accounting for both price and CO_2 signals. For instance, Sou *et al.* [26] propose a decision aiding framework for smart household appliances scheduling with the aim of finding a trade-off between minimization of electricity bill and CO_2 emissions. The start time of a dishwasher, a dryer, and a washing machine has to be assigned, while observing some constraints, such as user's preferred running interval. Similarly, Paridari *et al.* [24] apply mixed-integer linear programming techniques to solve a multi-objective optimization scheduling problem, which includes smart appliances, electrical storages, and use behavior uncertainties. Both studies use Swedish data for prices and CO_2 intensity, which appear to have a negative relation; as a consequence, a decrease in electricity bills corresponds to an in increase in emissions and vice versa. Nilsson *et al.* [22] draw analogous conclusions after investigating to what extent the visualization of spot prices, by means of a display installed in the stairwell of Swedish households, can affect residential electricity consumptions and can stimulate changes in consumption behavior. Braun *et el.* [5] propose the optimization of appliances in a residential and a commercial building, by solving a multi-objective problem that includes conflicting objectives, i.e., the minimization of costs, emissions, user discomfort, and technical wearout. Space conditioning control is the main objective of Dahl Knudsen *et al.* [9], which investigate the potential economic and environmental benefits of implementing a model predictive controller for Danish space heating system. Defined as a weighted sum of electricity costs and emissions, a purely economical oriented controller would reduce the consumptions in peak periods, but it would also cause an increase in CO_2 emissions. Recent studies [21, 19, 5] focus on the energetic and economic benefits of using hybrid appliances which integrates different energy carriers, such as electricity, natural gas, and hot water. The results show potential significant potential cost reductions, an increase in the efficiency and flexibility of the system. To the best of our knowledge, there are no reported studies that consider the use of CO_2 signals to schedule household appliances with respect to different energy carriers. Hybrid appliances are, indeed, still not extensively investigate, as their availability is limited [20].

Dynamic CO_2-equivalent intensity factor

[2] Given the generation mix of a power system and the hourly amount of energy generated per type of unit, it is possible to calculate the dynamic CO_2-equivalent intensity factor associated to one kWh of energy that an end-user buys from the grid. Stoll *et al.* present a method to calculate the hourly values based upon the amount of hourly electricity generated and traded, and the emission factors of the sources [28, 22]. In this study, we use the data available for the German electricity grid on the ENTSOE Transparency Platform [2]. Considering the emission factors in Table 1, the dynamic CO_2-equivalent intensity factor of the German grid varied in 2017 between 113 gCO_2eq/kWh and 533 gCO_2eq/kWh.

Emission factors for several kinds of sources are given in [29, 18, 10]; it is possible to quantify the emissions over all stages of electricity production, from production of infrastructure, technologies, and fuel, to conversion to electricity, to waste management. For the sake of completeness, we consider the emission values related to a life-cycle assessment. In particular, Weisser [29] reviews and compares the life-cycle GHG emissions for *present* and *future/advanced* technologies, the latter referring to best performance power plants with increased efficiency in a realistic 2010-2020 scenario. In Table 1, we report the emission factors used our studies. When available, we use the data for future/advanced technologies (i.e., for lignite, hard coal, gas, solar, and wind), otherwise mean values are considered. The areas in which improvements in GHG emissions may occur in future depend on the type of power plants. For instance, power plants burning fossil fuels are likely to be equipped with improved abatement technologies and burners; on the other hand, the development of new materials will reduce the emissions in the construction phase of wind turbines and solar panels [29].

According to the criteria used by ENTSOE in drawing its *Monthly Statistics Reports* [4], Table 1: Life-cycle GHG emission for power plants, expressed in gCO_2eq/kWh

Biomass	Solar PV	Wind onshore	Wind offshore	Geothermal	Pumped-Storage
71	43	8	9	45 *	34
Run-of-the-river	Reservoir	Nuclear	Lignite	Coal	Coal-derived gas
4	9	11	820	800	800
Gas	Oil	Waste	Other		
400	520	690 **	247		

The main source for these values is [29]. Values marked with * are taken from [10], and those marked with ** are from [17]. Since it is not specified which sources are included in "Other", the life-cycle GHG emission factor for this category is calculated as the mean value of all other known factors. We use the generation data available for the third Wednesday of four months of 2017, namely January, April, July, and October. Moreover, we include the data for May, 8th 2016, a sunny and windy Sunday when more than 50% of energy was produced by renewable plants, covering more than 90% of power demand in a couple of hours [8]. For each day, we calculate the dynamic CO_2-equivalent intensity factor for the German grid $EF_{e,t}$ as follows:

$$EF_{e,t} = \frac{\sum_{pp} E_{t,pp} \cdot EF_{pp}}{\sum_{pp} E_{t,pp}} \quad (1)$$

where $E_{t,pp}$ is the energy produced by power plant *pp* in time step *t*, and EF_{pp} is the life-cycle GHG emission factor of power plant *pp*.

[2] from [12]

Energy management system and operation scheduling

An Energy Management System coordinates the energy demand and supply between the generation units and the loads, while aiming at the fulfillment of economic and environmental objectives. The coordination can be implemented at various levels, from single household to larger portions of the grid, which grow in complexity and in interconnections among Distributed Energy Resources (DERs) and the grid. In our studies, the role of the EMS is to achieve the optimal scheduling of appliances, with the aim of minimizing the CO_2 emissions. In general, the operation scheduling is the planning of available resources, such as generators and storage, with the aim of minimizing operational costs and/or environmental impact in terms of emissions, while covering the energy demand. Where loads are shiftable or curtailable, they become part of the resources to be optimally planned, increasing system flexibility. In order to generalize the operation scheduling problem within the energy context, we propose a general definition that is specific for the planning of energy resources to satisfy the load demand, while being independent from the chosen model and objective functions.

Hybrid appliances

Residential appliances can be distinguished according to the energy carriers they require [21] and the operational characteristics [7]. Traditionally, household appliances use a single energy carrier to perform their function, usually electricity or gas, which cannot be substituted [21]. On the other hand, some devices can be supplied by multiple energy carriers, namely hot water, electricity, and gas, which are used alternatively or in parallel to operate [20]. This type of devices is referred to as *hybrid* and the potential benefit of their application have recently gained interest in literature [21, 19, 27]. Although the idea of suppling from different energy carriers may sound far from reality, there are solutions already commercially available. Some dishwashers and washing machine can be connected to a hot water supply, drastically reducing the power demand, as the water is no longer internally heated [27]. Gas clothes dryers heat the air by burning gas, while a hot water tumble dryer is equipped with a water-air heat exchanger [27]. Additionally, dual-fuel cookers are widely used, combining gas hobs and electric ovens; although not yet commercially available, it could be possible to combine electricity and gas in the same cooking device [20]. Moreover, water can be boiled in an electric kettle or in a kettle (or a pot) placed on an electric hob, as well as in a kettle placed on a gas hob [23]. The hot water needed for space heating and bathing can be provided by electric heaters as well as gas-burning boilers. An innovative alternative is being developed by Nerdalize [3], which uses cloud servers as preheating systems in homes, reducing at the same time the consumptions of gas for house heating and of electricity for server cooling.

Result

[3] Given the increasing electricity residential consumptions and the global attention towards GHG emissions, we propose a method to optimally schedule household loads according to CO_2 signals, with the aim of minimizing the daily equivalent carbon emissions of a single house. Hybrid appliances and the daily thermal load can use electricity imported from the grid, natural gas, or hot water produced in a gas-burning boiler, and, eventually, they can be shifted in time. The problem is formulated as a mixed integer linear problem, where the variables to be determined are binary and indicate the operation status of the appliances requiring a certain level of power or heat from the different energy carriers in each time step. The results show that the proposed algorithm allows to schedule the household loads by significantly reducing the carbon emissions. When the contribution of renewable sources in the generation mix is low, using multiple energy carriers, namely gas, hot water, and electricity, to supply the household appliances and the thermal load can reduce carbon emissions up to 30%, on a winter day. On the other hand, windy and sunny days can benefit more from a scheduling mainly based on electricity, although the maximum power importable from the

[3] mainly from [12]

grid can limit the carbon savings. This is a realistic condition for many European countries, such as Italy, where exceeding this limit leads to an automatic load shedding, and France, where users pay a subscription according to their expected maximum power demand. In general, shifting loads in time has a smaller effect; when the hybrid-mode is to be preferred, emission savings are marginally affected by the variability in the dynamic CO_2-equivalent intensity factor of the electrical grid, as the main energy carriers are hot water and gas, whose emission factor is constant. When the loads are mainly using electricity, shifting their operations in time while fulfilling the user's preference can reduce the emissions up to around 8%. Overall, switching from a single-carrier mode to a multi-carriers one and vice versa can successfully enable up to 74% equivalent CO_2 emissions reduction.

Use Case 1: household with hybrid appliances

[4] Our model includes six hybrid devices, namely the dishwasher, the washing machine, the tumble dryer, the oven, the hob, and the kettle. They can operate in two possible modes, i.e., electricity-only and hybrid; in the first one, all energy is supplied by the electricity grid and water/air is internally heated by means of an electric resistance. In the second one, DW, WM, and TD are connected to the hot water system and uses electricity for the basic functions of the machine (e.g., fun, motor, circulation pump, electronic devices etc). The cooker hob, the oven, and the kettle can either use only the electricity or they rely on natural gas as well as a small consumption of electricity mainly due to the ignition system. On the other hand, the space heating and hot water demand are combined and referred to as the thermal load. We assume that the thermal load is a single-energy-carrier load, which can be supplied either by a gas-burning boiler or an electric (water) heater system.

Additionally, the devices are distinguished in shiftable and on-demand. For the former type, the user selects a time window during which the device is supposed to run and conclude its pre-selected task, which corresponds to a power demand profile. The latter type includes not-shiftable devices that have to be switched on and off when needed.

In this context, the end-user makes two kind of sustainable choices: long-terms one, with respect to selecting the equipment for his household, and short-terms one. The latter includes, for instance, deciding hot to boil water, e.g., by using an electric kettle or with a tradition one on a gas stove, or choosing pre-heated hot water or cold water for running the dish washer or the washing machine. Moreover, the thermal load can by supplied by a gas boiler or an electric heater, which can be both available in a modern house.

Use Case 2: cooking pasta

In our everyday lives we constantly, often implicitly take sustainability decisions. In 2017, cooking a dish of pasta occurred on a daily basis for 63% of the Italian population. When taking a pot to the sink to fill it with water for boiling the pasta, the person cooking is faced with the choice of starting with either hot or cold water. People may just take water without thinking, or they could be using hot water with the intention of speeding up the process. Which is the environmental impact of such a choice? Of course, this is just one example, but we make many of such decisions on a daily basis. Do we walk the stairs or take the elevator? Do we set the room temperature to a couple of degrees more or not? Being aware of each of them and accounting for their environmental impact is hardly feasible. First, it is unclear how to trace back a decision to its sustainability footprint; second, the complexity of the implications of all such decisions is overwhelming; third, our cognitive resources are too limited to do so. The burden of sustainability decisions can be reduced by the growth in digitalization and automation systems, which can support the user in reducing the environmental

[4] mainly from [12]

impact of her daily choices. Nevertheless, some questions are still open. Do people accept automated systems and under which conditions? Which factors could enhance the acceptability of an automated system in everyday activities? Would the user accept to adjust the time and the way of cooking pasta to increase the efficiency of the process? For our study, we measure sustainability in terms of CO_2 emissions and we see that, for households for which water is heated with gas and cooking is electric, starting from hot or cold water when cooling a dish of pasta may mean a difference in emission of up to four times.

Equipment	Efficiency %	Task	Energy consumption Wh/100 ml
Gas boiler	95	Preheating	
Electric heater	100	Preheating	
Electric kettle	95	Boiling	10.4 [23]
Microwave oven	63	Boiling	20.7 [23]
Ceramic hob	65	Boiling/cooking	15.3 [23]
Solid hot plate	70	Boiling/cooking	14.2 [6]
Induction hob	85 [13]	Boiling/cooking	11.6
Gas hob	52 [13]	Boiling/cooking	19.0

Table 2: Efficiency factors of the equipment involved in the process of cooking pasta.

According to Italian cookbooks and tradition, cooking pasta entails using 1 liter of water per 100 g of dried pasta, which is a generous amount for one person. The water has to be brought to 100° C, and then the pasta can be thrown in. Depending on the shape, freshness, and thickness of the pasta, the water with the pasta has to be kept boiling for anywhere between few minutes to a quarter of an hour.

Several options are possible for bringing the water from room temperature to its boiling temperature and then cooking the pasta: using a pot on ceramic hob with radiant heating, a kettle, a microwave, a gas stove, or a solid hotplate. We size the model assuming 5-6 people portion, that is, boiling 5 liters of water in order to cook 500 g of dried pasta. There are two phases in the process: (1) bringing the water to its boiling temperature; and (2) keeping it boiling to cook the pasta. Several devices can be used in this process, whose energy consumptions for boiling water and efficiency factors are summarized in Table 2.

Consider a specific day in Germany, say 17 June 2018 and a user desiring to cook half kilo of pasta. She can decide to do it during breakfast, and they will eat it later for lunch; she can cook it around lunch time; or she can cook it for dinner. We consider three emission factors, i.e., 330 gCO_2/kWh, 240 gCO_2/kWh, and 380 gCO_2/kWh, which correspond to the EF at 6 am, 12pm, and 6.30 pm, respectively. Moreover, we consider that she uses the best equipment configuration, that is, she boils the water using the kettle and she cooks the pasta using an induction hob. Moreover, she may have a gas boiler or an electric heater. The most sustainable choice is to cook pasta at 12pm, preheating the water with an electric heater as shown in Table 3. Besides a difference in emissions of around 60% between lunch time and dinner time, when the electricity EF has the lowest and highest value, respectively, it is worth noticing that preheating the water is usually the most sustainable choice. However, at lunch time, when the EF is low, using a gas boiler for the preheating would increase the emissions by 5%.

Table 3: Cooking pasta on 17 June 2018

	EF=330	EF=240	EF=380

EE	272	198	313
GEE	263 (-3%)	208 (+5%)	293 (-6%)
EEE	268 (-1%)	195 (-1%)	309 (-1%)

The modeling of a simple daily task such as cooking has shown that a simple choice like using hot or cold tap water is also an implicit environmental choice. Depending on the energy-mix of the moment and chosen equipment, CO_2 emissions can vary significantly. In our model, it is shown to be as high as 30%.

It is worth remarking that, while the amount of CO_2 savings per event might look small (i.e., 50 g of CO_2 by preheating the water for 500 g of pasta), the combined effect of behavior change of a community will be significant. Considering that an average tree in central Europe absorbs around 10 kg of CO_2 per year [1], 2000 families cooking pasta three times per week and following sustainable choices could reduce as much CO_2 as a hectare of trees would absorb in one year by just preheating water.

The many factors involved in the model imply that it is too complicated for user to be constantly aware of and be able to consider it. Fortunately, with the growth in digitalization, personal assistants, and home automation system, one can support the user in making sustainable choices throughout the day. When reading a recipe off the Web, the home automation system may recommend to use cold water to execute it or the user could proactively ask an app how to best proceed environmentally. However, would the people accept and proactively use such system? Would they feel frustrated by the complexity of those decisions? These and more questions on the user acceptability of automated systems and on the trade-off between control and automation should be further investigated.

References

[1] Carbon sequestration by biomeiler wood compost. http://biomeiler.nl/carbonsequestration-by-biomeiler-wood-compost/.

[2] ENTSO-E Transparency Platform. https://transparency.entsoe.eu/.

[3] Nerdalize https://www.nerdalize.com

[4] Statistics and data. https://electricity.network-codes.eu/publications/statistics-anddata/

[5] Marlon Braun, Thomas Dengiz, Ingo Mauser, and Hartmut Schmeck. Comparison of Multi-objective Evolutionary Optimization in Smart Building Scenarios. In European Conference on the Applications of Evolutionary Computation, pages 443–458. Springer, Cham, 2016.

[6] Annika Carlsson-Kanyama and Kerstin Bostr¨om-Carlsson. Energy Use for Cooking and Other Stages in the Life Cycle of Food - A study of wheat, spaghetti, pasta, baley, rice, potatoes, couscous and mashed potatoes, volume 01. 2001.

[7] Zhi Chen, Lei Wu, and Yong Fu. Real-time price-based demand response management for residential appliances via stochastic optimization and robust optimization. IEEE Transactions on Smart Grid, 3(4):1822–1831, 2012.

[8] Craig Morris. Germany nearly reached 100 percent renewable power on Sunday.

[9] Michael Dahl Knudsen and Steffen Petersen. Demand response potential of model predictive control of space heating based on price and carbon dioxide intensity signals. Energy and Buildings, 125:196–204, 2016.

[10] Ottmar Edenhofer, Ram´on Pichs-Madruga, Youba Sokona, Kristin Seyboth, Patrick Eickemeier, Patrick Matschoss, Gerrit Hansen, Susanne Kadner, Steffen Schl¨omer, Timm Zwickel, and Christoph Von Stechow. IPCC, 2011: Summary for Policymakers. In: IPCC Special Report on Renewable Energy Sources and Climate Change Mitigation. 2011.

[11] European Commission. Buildings.

[12] Laura Fiorini and Marco Aiello. Household CO 2 -efficient energy management. Energy Informatics, 1(Suppl 1):21–34, 2018.

[13] Tiffany J. Hager and Ruben Morawicki. Energy consumption during cooking in the residential sector of developed nations: A review. Food Policy, 40:54–63, 2013.

[14] Victoria Haines, Kevin Lomas, Thomson Murray, Richardson Ian, Nhamra Tracy, and Giulietti Monica. How Trends in Appliances Affect Domestic CO2 Emissions : A Review of Home and Garden Appliances. Department of Energy and Climate Change, (March 2010):1–6, 2010.

[15] IEA. Energy and Climate Change. World Energy Outlook Special Report, pages 1–200, 2015.

[16] IPCC. Climate Change 2013, volume 5. 2014.

[17] B Johnke. Emissions From Waste Incineration. Good Practice Guidance and Uncertainty Management in National Greenhouse Gas Inventories, pages 455–468, 2009.

[18] Jun Kono, York Ostermeyer, and Holger Wallbaum. The trends of hourly carbon emission factors in Germany and investigation on relevant consumption patterns for its application. International Journal of Life Cycle Assessment, 22(10):1493–1501, 2017.

[19] Ingo Mauser, Jan Mu¨ller, Florian Allerding, and Hartmut Schmeck. Adaptive building energy management with multiple commodities and flexible evolutionary optimization. Renewable Energy, 87:911–921, 2016.

[20] Ingo Mauser, Jan Mu¨ller, and Hartmut Schmeck. Utilizing Flexibility of Hybrid Appliances in Local Multi-modal Energy Management. In Proceedings of the 9th international conference on energy efficiency in domestic appliances and lighting (EEDAL ’17), number Section 3, pages 1282–1297, Irvine, California, USA, 2017.

[21] Ingo Mauser, Hartmut Schmeck, and Uwe Schaumann. Optimization of Hybrid Appliances in Future Households. Die Energiewende - Blueprints for the new energy age; Proceedings of International ETG Congress 2015;International ETG Congress 2015;, pages 165–170, 2015.

[22] Anders Nilsson, Pia Stoll, and Nils Brandt. Assessing the impact of real-time price visualization on residential electricity consumption, costs, and carbon emissions. Resources, Conservation and Recycling, 124:152–161, 2017.

[23] Claudia Oberascher, Rainer Stamminger, and Christiane Pakula. Energy efficiency in daily food preparation. International Journal of Consumer Studies, 35(2):201–211, 2011.

[24] Kaveh Paridari, Alessandra Parisio, Henrik Sandberg, and Karl Henrik Johansson. Robust Scheduling of Smart Appliances in Active Apartments With User Behavior Uncertainty. IEEE Transactions on Automation Science and Engineering, 13(1):247–259, 2015.

[25] Pierluigi Siano. Demand response and smart grids - A survey. Renewable and Sustainable Energy Reviews, 30:461–478, 2014.

[26] Kin Cheong Sou, K Mikael, Jonas Wu, Henrik Sandberg, and Karl Henrik Johansson. Energy and CO 2 Efficient Scheduling of Smart Home Appliances. pages 4051–4058, 2013.

[27] R. Stamminger. Synergy Potential of Smart Appliances. D2.3 of WP2 from the Smart-A project, page 237, 2008.

[28] Pia Stoll, Nils Brandt, and Lars Nordstr¨om. Including dynamic CO2 intensity with demand response. Energy Policy, 65:490–500, 2014.

[29] Daniel Weisser. A guide to life-cycle greenhouse gas (GHG) emissions from electric supply technologies. Energy, 32(9):1543–1559, 2007.

Distributed Energy Storage System

Generally, manufacturing assets, especially tangible assets (i.e., factories and warehouses) consume electrical energy to produce products. Therefore, the electrical energy supply is an essential problem for the manufacturing asset. The generation and distribution of electrical energy are commonly done in the traditional manner where energy is centrally generated and delivered to all end-users. However, this traditional method of energy supply to the manufacturing asset is currently subjected to innovation with a focus on decentralized energy generation combined with the Distributed Energy Storage System (DESS). A new and smarter way of energy supply to the manufacturing asset, called a "Smart Grid", is likely to hold the future. The Smart Grid is an electricity network that uses Information and Communication Technology (ICT) to integrate renewable energy into the power grid, and enhance the efficiency and reliability of the power grid. Unlike the traditional method, the central energy provider, like the utility company, is no longer the most important component for supplying energy, rather the end-users in the grid itself are. This implicates that the manufacturing asset can generate electrical energy itself through renewable energy generators such as wind turbines or solar panels, as well as receive energy from the central energy provider or other manufacturing assets in the grid. The manufacturing asset that both produce and consume electrical energy is called a "Prosumer".

This trend of increasingly using renewable energy in the Smart Grid, however promising, faces a major challenge as well. A prosumer with a lot of solar panels will produce more energy on a sunny day than it consumes, and therefore will want to sell its surplus of energy. In the evening, when there is no sun, at the level of a prosumer production is lower than the consumption which incites the prosumer to buy any lacking energy. From this example, one can observe that there is no continuous integration of energy generation. Furthermore, there is no guarantee that the prosumer can sell its excess energy to other prosumers in the grid. The solution to this problem lies in bridging the energy gap between day and night by storing the surplus of energy.

The Distributed Energy Storage System requires the conversion of electricity into another form of energy that can be stored, which later is converted back again into electricity. The focus of our work is to introduce the battery-based DESS into manufacturing assets to stabilize electrical energy flows and improve the efficiency of energy usage, and examine the economic benefit of investment of installing the battery-based DESS and renewable energy generators. Hence, the question addressed in this work is: "What is the effect of introducing the battery-based Distributed Energy Storage Systems into the manufacturing assets?"

In this work, we model manufacturing assets with renewable energy generators (wind turbines and solar panels) and battery-based DESSs. We also model the strategy of using the battery-based DESSs. Then, we build a simulation program to examine our models.

Definitions

A battery-based DESS can be charged up in advance to store surplus energy and be discharged when the prosumer has a lack of energy. These new states should be taken into account when running the simulation. At every decision, a choice should be made based on which option is most beneficial at that moment; storing, discharging, selling or buying energy. Therefore, we introduce a decision tree in regard to the strategy for the battery-based DESS. Relevant parameters in this decision tree are stated below.

Let Q_i be the input energy where $Q_i = |E_{produced} - E_{consumed}|$.
Let $\eta_c \subset (0, 1]$ denote the charging efficiency of the battery.
Let $\eta_d \subset (0, 1]$ denote the discharging efficiency of the battery.

Let $\eta_r = \eta_d \times \eta_c$ denote the round-trip efficiency of the battery.
Let $SoC_{i,max}$ denote the upper limit of the state of charge, energy capacity (kWh).
Let $SoC_{i,min}$ denote the lower limit of the state of charge (kWh).
Let $S_{i,max} = SoC_{i,max} - SoC_{i,min}$ denote the max amount of energy the battery can store.
Let S_i denote the amount of energy that the battery can discharge.
Let P_{Si} denote the average price of the energy in the battery.
Let α_{dis} denote the amount of energy provided by the battery.
Let α_{buy} denote the amount of energy bought from other prosumers.
Let CoDis(x) denote the cost of discharging the the battery with amount of x.
Let $LoD(Q_i)$ be the loss of delivery from energy Q_i.
Then, we have following expressions:
$CoDis(S_i) = S_i \times P_{Si}$
$(1 - \eta_r) \times Q_i$ is the energy loss of storing up Q_i.
$S_{i,max}$ S_i is the amount of energy that can be stored.
η_d S_i is the energy can be discharged from the battery.
$Q_i / \eta_d \times P_{Si}$ is its cost from the amount of discharged energy is Q_i.
$CoDis(S_i) + CoB(Q_i - \eta_d S_i)$ is the energy cost, when the demand can be partially satisfied by discharging the battery.

The following conditions and assumptions are made in this model; we assume that there is no degradation in charging/discharging performance and energy capacity of the battery. We assume as well that there is no storage loss in the battery. The new strategy of using the battery can be visualized as a decision tree shown below.

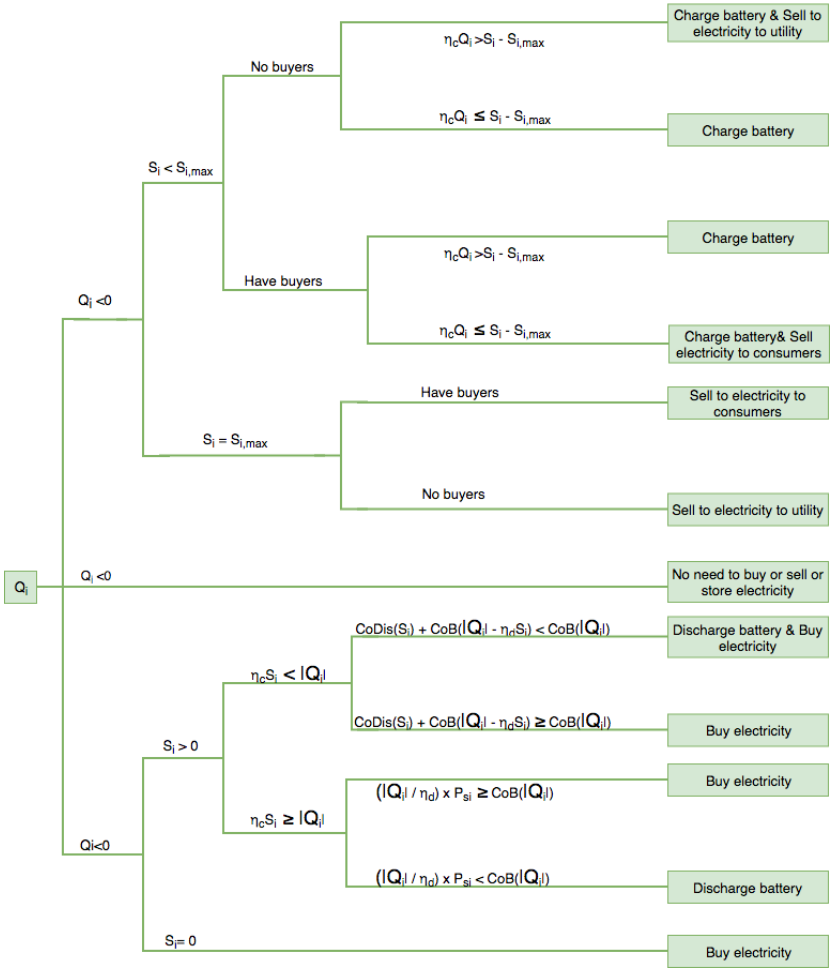


Figure x.1: Decision tree to decide the end state of the energy

To get prosumers to actually buy a battery, there should be benefits for buying it for them. On the one side, a decrease in energy cost would be favourable, on the other side, a decrease in energy loss would be beneficial to the environment. This optimization problem for using the battery can be described as:

$$\begin{cases} min(CoB(\alpha_{buy}) + CoDis(\alpha_{dis})) & (Q_i < 0) \\ min((1-\eta_r) \times Q_i, LoD(Q_i) & (Q_i > 0)) \end{cases}$$

Where $\alpha_{buy} + \alpha_{dis} = Q_i$

For a prosumer, the surplus energy has to be sold or stored. The lack of energy has to be compensated by discharging the battery-based DESS or buy the energy from someone else in the grid. This process can be divided in two states, namely in the decision state and in the action state. For every prosumer in each time slot we will look at the state of that prosumer. If the prosumer has for example surplus energy, the actions are cut down to only selling or storing. The decision is made based on information of the prosumer and his corresponding battery-based DESS. The most important parameters are the amount of energy that is in the battery (S_i) and the amount of surplus energy (Q_i). The decision will be made with the decision tree in Figure x.1. After the decision is made it will be send to both the prosumer and the corresponding battery-based DESS. Both will know the decision that is made, and they will react to it by performing the right actions. This decision process for a prosumer in the surplus energy state can be seen in the sequence diagram below.

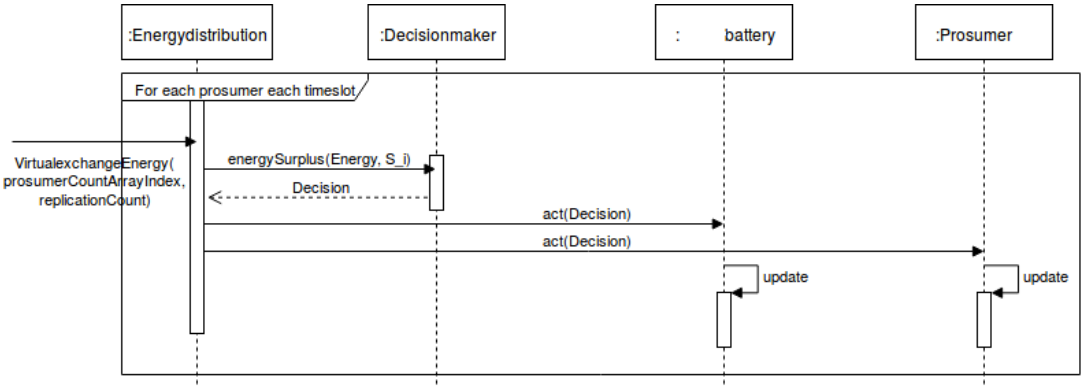


Figure x.2: Sequence diagram: Decision making process where the prosumer has surplus energy

Experiments

One simulation can be regarded as a day with 24 time slots, and one time slot is 1 hour. The simulation runs 2000 times, which resembles approximately 5.5 years.

An important parameter for making decisions is the pricing of energy. Here we will discuss the prices that were used in this simulation. We calculated the Cost of Storage (CoS) of a battery, the Cost of Generation (CoG) regarding solar panels and wind turbines.

Cost of Storage is the price of storing energy in the battery-based DESS. The Tesla Power Wall 1 is taken as an example for the correct values. The cost of a Tesla Power Wall 1 is 3000 $ = 2,687.1 EUR. The capacity of a Tesla Power wall 1 is 6.4 kWh per full cycle. For the amount of cycles (charge/discharge count) we consider the 5,000 cycles under warranty. Then, we have CoS = 0.1 EUR/kWh.

For the solar panel, we use Jinko Solar JKM 270PP-60 Eagle Solar, which is sold for 0.484 EUR/Wp. However, a 1000 Wp solar array will need an inverter to convert the direct current produced by the solar panels into alternating current. These inverters cost approximately 200 EUR/1000Wp. Installation costs and materials (rails, clamps, cables) are calculated at 216 EUR. Higher and lower prices will depend on the market. Most solar panels have a lifetime of at least 20 years. Beyond 20 years of existence, the solar panel installation is paid off completely, and the cost of generation that is still calculated in our simulation will be a profit for the owner. A 1000 Wp solar array will yield an average of 900 kWh per year. For the wind turbine, we sue Bergey BWC Excel S50 which is approximately 2000 EUR/kW. Then, we have average CoG = 0.06 EUR/kWh.

The results of simulation is shown below, where only the cost of storage is increased from 0 cent to 20 cent/kWh, and the cost of generation is kept unchanged at 0.06 EUR/kWh. The figure below shows that a battery is actually in use because the battery is discharged approximately 13.7 times per day, only while the cost of storage is lower than 0.16 EUR/kWh. In the same figure we see that if the cost of storage is 0.16 EUR/kWh or higher, the discharge counter drops to zero discharges per day. Because the cost of generation is set at 0.06 EUR/kWh in our simulation, and the total cost of energy stored in the battery which is the sum of (CoS + CoG) the calculation will be CoS (= 0.16 EUR/kWh) + CoG (wind,solar = 0.06 EUR/kWh), equals the utility price of 0.22 EUR/kWh, the discharge counter drops to zero. At this point the utility price equals (CoS + CoG), and discharging the battery is not beneficial any more.

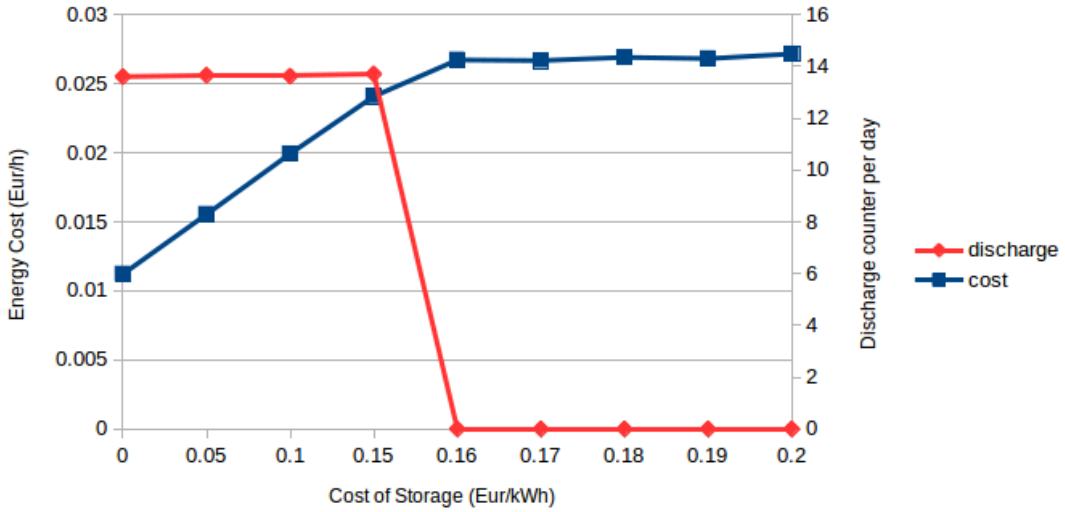


Figure x.3: Discharge Counter with mutiple values of the Cost of Storage

For this experiment, the Tesla power wall 1 is used for the simulation. However, other battery models can be easily incorporated into the simulation as well. The effect of the charge/discharge efficiency of a battery is not taken into account in this simulation. An improvement can be done by including this effect in the simulation for more precise results. As can be seen in these promising results, introducing a battery-based DESS to manufacturing assets has clear benefits on multiple accounts and is likely to hold the future.