3

Reference Architecture for Factory Automation using Edge Computing and Blockchain Technologies

Mauro Isaja

Engineering Ingegneria Informatica SpA, Italy
E-mail: mauro.isaja@eng.it

This chapter will introduce the reader to the FAR-EDGE Reference Architecture (RA): the conceptual framework that, in the scope of the FAR-EDGE project, was used as the blueprint for the proof-of-concept implementation of a novel *edge computing platform for factory automation*: the FAR-EDGE Platform. Such platform is going to prove edge computing's potential to increase flexibility and lower costs, without compromising on production time and quality. The FAR-EDGE RA exploits best practices and lessons learned in similar contexts by the global community of system architects (e.g., Industrie 4.0, Industrial Internet Consortium) and provides a terse representation of concepts, roles, structure and behaviour of the system under analysis. Its unique approach to edge computing is centered on the use of *distributed ledger technology (DLT)* and *smart contracts* – better known under the collective label of *Blockchain*. The FAR-EDGE project is exploring the use of Blockchain as a key enabling technology for industrial automation, analytics and virtualization, with validation use cases executed in real-world environments that are briefly described at the end of the chapter.

3.1 FAR-EDGE Project Background

FAR-EDGE's main goal is to provide a novel edge computing solution for the virtualization of the factory automation pyramid. The idea of decentralizing factory automation is not new. Rather, for over a decade, several initiatives, including background projects of the consortium partners, have introduced decentralized factory automation solutions based on various technologies like intelligent agents and service-oriented architectures (SOA). These background initiatives produced proof-of-concept implementations that highlighted the benefits of decentralized automation in terms of flexibility; yet they are still not being widely deployed in manufacturing plants. Nevertheless, the vision is still alive, as this virtualization can make production systems more flexible and agile, increase product quality and reduce cost, e.g., enable scalable, fast-configurable production lines to meet the global challenges of *mass-customization* and *reshoring*.

With the advent of the Industrie 4.0 and the Industrial Internet of Things (IIoT), such solutions are revisited in the light of the integration of Cyber-Physical Systems (CPS) within cloud computing infrastructures. Therefore, several cloud-based applications are deployed and used in factories, which leverage the capacity and scalability of the cloud, while fostering supply chain collaboration and virtual manufacturing chains. Early implementations have also revealed the limitations of the cloud in terms of efficient bandwidth usage and its ability to support real-time operations, including operations close to the field. In order to alleviate these limitations, edge computing architectures have recently introduced. Edge computing architectures introduce layers of edge nodes between the field and the cloud, as a means of:

- **Saving bandwidth and storage**, as edge nodes can filter data streams from the field in order to get rid of information that does not provide value for industrial automation.
- **Enabling low-latency and proximity processing**, since information can be processed close to the field, rather in a remote (back-end) cloud infrastructure.
- **Providing enhanced scalability**, given that edge computing supports decentralized storage and processing that scale better when compared to conventional centralized cloud processing. This is especially the case when interfacing to numerous devices is needed.
- **Supporting shopfloor isolation and privacy-friendliness**, since edge nodes deployed at the shopfloor can be isolated from the rest of the edge network. This can provide increased security and protection of manufacturing dataset in cases required.

These benefits make edge computing suitable for specific classes of use cases in factories, including:

- **Large-scale distributed applications**, typically applications that involve multiple plants or factories, which collect and process streams from numerous distributed devices in large scale.
- **Nearly real-time applications**, which need to analyze data close to the field or even control CPS such as smart machines and industrial robots. A special class of such real-time applications involves edge analytics applications.

As a result, the application of edge computing for factory automation is extremely promising, since it can support decentralized factory automation in a way that supports real-time interactions and analytics in large scale. FAR-EDGE researches have explored the application of the edge computing paradigm in factory automation, through designing and implementing reference implementations in line with recent standards for edge computing in industrial automation applications. Note that FAR-EDGE was one of the first initiatives to research and experiment with edge computing in the manufacturing shopfloor, as relevant activities were in their infancy when FAR-EDGE project was approved. However, the state of the art in factory automation based on edge computing has evolved and FAR-EDGE efforts are taking into account this evolution.

3.2 FAR-EDGE Vision and Positioning

FAR-EDGE's vision is to research and provide a proof-of-concept implementation of an *edge computing platform for factory automation*, which will prove edge computing's potential to increase automation flexibility and lower automation costs, without however compromising production time and quality. The FAR-EDGE architecture is aligned to the IIC RA, while exploiting concepts from other RAs and standards such as the OpenFog RA and RAMI 4.0 (see below for more details). Hence, the project will be providing one of the world's first reference implementation of edge computing for factory automation. Within this scope, FAR-EDGE will offer a host of functionalities that are not addressed by other implementations, such as IEC-61499 compliant automation and simulation.

Beyond its functional uniqueness, FAR-EDGE is also unique from a research perspective. In particular, the project is researching the applicability of disruptive KETs: *distributed ledger technology (DLT)* and *smart contracts* – better known under the collective label of *Blockchain*. The Blockchain concept, while being well understood and thoroughly tested in mission-critical areas like digital currencies (e.g., Bitcoin, Ethereum), has never been applied before to industrial systems. FAR-EDGE aims at demonstrating how a pool of services built on a generic Blockchain platform can enable decentralized factory automation in an effective, reliable, scalable and secure way. In FAR-EDGE, such services are responsible for sharing process state and enforcing business rules across the computing nodes of a distributed system, thus permitting virtual automation and analytics processes that span multiple nodes – or, from a bottom-up perspective, autonomous nodes that cooperate to a common goal.

3.3 State of the Art in Reference Architectures

A *reference architecture* (RA) is often a synthesis of best practices having their roots in past experience. Sometimes it may represent a "vision", i.e., a conceptual framework that aims more at shaping the future and improving over state-of-the-art design rather than at building systems faster and with lower risk. The most successful RAs – those that are known and used beyond the boundaries of their native ground – are those combining both approaches. Whatever the strategy, an RA is for teamwork: its major contribution to development is to set a common context, vocabulary and repository of patterns for all stakeholders.

In FAR-EDGE, where we explore the business value of applying innovative computing patterns to the smart factory, starting from an effective RA is of paramount importance. For this reason, the FAR-EDGE Reference Architecture was the very first outcome of the project's platform development effort.

In our research, we considered some well-known and accepted *generic RAs* (see sub-section below) as sources of inspiration. The goal was twofold: on the one hand, to leverage valuable experience from large and respected communities; on the other hand, to be consistent and compatible with the mainstream evolution of the smart factory, e.g., Industrial IoT and Industry 4.0. At the end of this journey, we expect the FAR-EDGE RA to become an asset not only in the scope of the project (as the basis for the FAR-EDGE Platform's design), but also in the much wider one of factory automation, where it may guide the design of ad-hoc solutions having edge computing as their main technology driver.

3.3.1 Generic Reference Architectures

A generic RA is one that, while addressing a given field of technology, is not targeting any specific application, domain, industry or even (in one case) sector. Its value is mainly in communication: lowering the impedance of information flow within the development team and possibly also towards the general public. As such, it is basically an ontology and/or a mind mapping tool. However, as we will see further on in this analysis, sometimes the ambition of a generic RA is also to set a standard for runtime interoperability of systems and components, placing some constraints on implementation choices. Obviously, for this approach to make sense, it should be backed by a critical mass of solution providers, all willing to give up the vendor-lock-in competitive factor in exchange for the access to a wider market.

We have identified three generic RAs that have enough traction to influence the "technical DNA" of the FAR-EDGE Platform: RAMI 4.0, IIRA and OpenFog RA. In the following sub-sections, each of them is briefly analysed and, when it is the case, some elements that are relevant to FAR-EDGE are extracted to be reused later on.

3.3.2 RAMI 4.0

The Reference Architectural Model for Industrie 4.0 (RAMI 4.0)[1] is a generic RA addressing the manufacturing sector. As its name clearly states, it is the outcome of Platform Industrie 4.0,[2] the German public–private initiative addressing the fourth industrial revolution, i.e., merging the digital, physical and biological worlds into CPS.

According to some experts [1], the expected benefits of the adoption of CPS in the factory are:

- higher quality
- more flexibility
- higher productivity
- standardization in development
- products can be launched earlier
- continuous benchmarking and improvement
- global competition among strong businesses
- new labour market opportunities
- creation of appealing jobs at the intersection of mechanical engineering, automation and IT
- new services and business model

---

[1] https://www.zvei.org/en/subjects/industry-4-0/

[2] http://www.plattform-i40.de/I40/Navigation/EN/Home/home.html

To ensure that all participants involved in discussions understand each other, RAMI 4.0 defines a 3D structure for mapping the elements of production systems in a standard way.

RAMI 4.0, however, is also a standard-setting effort. While still a work in (slow) progress at the time of writing [2], its roadmap includes the definition of a globally standardized communication architecture that should enable the plug-and-play of *Things* (e.g., field devices, connected factory tools and equipment, smart machines, etc.) into composite CPS. Currently, only the general concept of *I4.0 Component* has been introduced: any Thing that is wrapped inside an *Administration Shell*, which provides a standard interface for communication, control and management while hiding the internals of the actual physical object. Future work will identify standard languages for the exchange of information, define standard data and process models and include recommendations for implementation – communication protocols in the first place.

With respect to the latter point, OPC UA is central to the RAMI 4.0 strategy. It is the successor of the much popular (in Microsoft-based shopfloors) OPC machine-to-machine communication protocol for industrial automation. As opposed to OPC, OPC UA is an open, royalty-free cross-platform and supports very complex information models. I4.0 Components will be required to adopt OPC UA as their interfacing mechanism, while also relying on several IEC standards (e.g., 62832, 61804, etc.) for information sharing.

RAMI 4.0 has gained a significant traction in Germany and is also driving the discussion around Industry 4.0 solutions and platforms in Europe. In particular, its glossary and its 3D structure for element mapping are increasingly used in sector-specific projects (in particular, platform-building ones) and working groups as a common language. The FAR-EDGE RA will adopt some of the RAMI 4.0 conceptual framework as its own, simplifying communication with the external communities of developers and users.

3.3.3 IIRA

The Industrial Internet Reference Architecture (IIRA)[3] has been developed and is actively maintained by the Industrial Internet Consortium (IIC), a global community of organizations (>250 members, including IBM, Intel, Cisco, Samsung, Huawei, Microsoft, Oracle, SAP, Boeing, Siemens, Bosch and General Electric) committed to the wider and better adoption of the

---

[3] http://www.iiconsortium.org/IIRA.htm

Internet of Things by the industry at large. The IIRA, first published in 2015 and since evolved into version 1.8 (Jan 2017), is a standards-based architectural template and methodology for the design of Industrial Internet Systems (IIS). Being an RA, it provides an ontology of IIS and some architectural patterns, encouraging the reuse of common building blocks and promoting interoperability. It is worth noting that a collaboration between the IIC and Platform Industrie 4.0, with the purpose of harmonizing RAMI 4.0 and IIRA, has been announced.[4]

IIRA has four separate but interrelated *viewpoints*, defined by identifying the relevant stakeholders of IIoT use cases and determining the proper framing of concerns. These viewpoints are: business, usage, functional and implementation.

- The *business viewpoint* attends to the concerns of the identification of stakeholders and their business vision, values and objectives. These concerns are of particular interest to decision-makers, product managers and system engineers.
- The *usage viewpoint* addresses the concerns of expected system usage. It is typically represented as sequences of activities involving human or logical users that deliver its intended functionality in ultimately achieving its fundamental system capabilities.
- The *functional viewpoint* focuses on the functional components in a system, their interrelation and structure, the interfaces and interactions between them and the relation and interactions of the system with external elements in the environment.
- The *implementation viewpoint* deals with the technologies needed to implement functional components, their communication schemes and their life cycle procedures.

In FAR-EDGE, which deals with platforms rather than solutions, the functional and implementation viewpoints are the most useful.

The functional viewpoint decomposes an IIS into functional domains, which are, following a bottom-up order, *control*, *operations*, *information*, *application* and *business*. Of particular interest in FAR-EDGE are the first three.

The *control domain* represents functions that are performed by industrial control systems: reading data from sensors, applying rules and logic and exercising control over the physical system through actuators. Both accuracy and

---

[4] http://www.iiconsortium.org/iic-and-i40.htm – to date, no concrete outcomes of such collaboration have been published.

resolution in timing are critical. Components implementing these functions are usually deployed in proximity to the physical systems they control, and may therefore be distributed.

The *operations domain* represents the functions for the provisioning, management, monitoring and optimization of the systems in the control domain.

The *information domain* represents the functions for gathering and analysing data to acquire high-level intelligence about the overall system. As opposed to their control domain counterparts, components implementing these functions have no timing constraints and are typically deployed in factory or corporate data centres, or even in the cloud as a service.

Overall, the functional viewpoint tells us that control, management and data flow in IIS are three separate concerns having very different non-functional requirements, so that implementation choices may also differ substantially.

The implementation viewpoint describes some well-established architectural patterns for IIS: the Three-tier, the Gateway-mediated Edge Connectivity and Management and the Layered Databus. They are of particular interest in FAR-EDGE, as they all deal with edge computing, although in different ways.

The *Three-tier architectural pattern* distributes concerns to separate but connected tiers: Edge, Platform and Enterprise. Each of them play a specific role with respect to control and data flows. Consistently with the requirements stemming from the functional viewpoint, control functionality is positioned in the Edge Tier, i.e., in close proximity to the controlled systems, while data-related (information) and management (operations) services are part of the Platform. However, the IIRA document v1.8 also states that in real systems, some functions of the information domain may be implemented in or close to the edge tier, along with some application logic and rules to enable *intelligent edge computing*. Interestingly enough, though, the opposite – edge computing as part of Platform functionality – is not contemplated by IIRA, probably because intelligent edge nodes (i.e., connected factory equipment with on-board computing capabilities) are deemed to be an OEM's (Original Equipment Manufacturer) concern. However, there is a component in the IIRA diagram suggesting that such boundaries may be blurred: the Gateway, which is part of the Edge Tier, connects it to both the Platform and Enterprise ones.

The Edge Gateway (EG) is in fact the focus point of another IIRA architectural pattern: the *Gateway-mediated Edge Connectivity and Management*. It allows for localizing operations and controls (edge analytics and computing). Its main benefit is in breaking down the complexity of the IIS, so that it may scale up in both numbers of managed assets and networking. The EG acts as an endpoint for the wide-area network while isolating the individual local networks of edge nodes. It may be used as a management point for devices and as an aggregation hub where some data processing and control logic is deployed.

The implementation viewpoint indeed provides some very relevant building blocks for the FAR-EDGE platform. What we see as a gap in the IIRA approach, up to this point, is the lack of such a block for addressing *distributed computing,* which is implied in the very notion of edge computing when used as a load-distribution technique for systems that are still centralized in their upper tiers. A partial answer to this question is given by the third and last IIRA architectural pattern: the *Layered Databus*. According to this design, an IIS can be partitioned into multiple horizontal layers that together define a hierarchy of scopes: machine, system, system of systems and Internet. Within each layer, components communicate with each other in a *peer-to-peer* (P2P) fashion, supported by a layer-specific databus. A databus is a logical connected space that implements a common data model, allowing interoperable communications between endpoints at that layer. For instance, a databus can be deployed within a smart machine to connect its internal sensors, actuators, controls and analytics. At the system level, another databus can be used for communications between different machines. At the system of systems level, still another databus can connect together a series of systems for coordinated control, monitoring and analysis.

In FAR-EDGE, the concept of cross-node P2P communication is going to play a key role as the enabling technology for edge computing in the three functional domains of interest: control, operations and information.

3.3.4 OpenFog RA

The OpenFog Consortium[5] is a public–private initiative, which was born in 2015 and shares similarities to the IIC: both consortia share big players like IBM, Microsoft, Intel and Cisco as their founding members and both use the ISO/IEC/IEEE 42010:2011 international standard[6] for communicating architecture descriptions to stakeholders. However, the OpenFog initiative is not constrained to any specific sector: it is a technology-oriented ecosystem that fosters the adoption of *fog computing* in order to solve the bandwidth, latency

---

[5] https://www.openfogconsortium.org/

[6] https://www.iso.org/standard/50508.html

and communications challenges of IoT, AI, robotics and other advanced concepts in the digitized world. Fog computing is a term first introduced by Cisco, and is basically a synonym for edge computing[7]: both refer to the practice of moving computing and/or storage services towards the edge nodes of a networked system.

The OpenFog RA was first released at the beginning of 2017, and as such it is the most recent contribution to the mainstream world of IoT-related architectures. The technical paper that describes it[8] is quite rich in content. As in IIRA, *viewpoints* are used to frame similar concerns, which in OpenFog RA are restricted to *functional* and *deployment* (the latter being roughly equivalent of IIRA's *implementation* viewpoint). However, these topics are not discussed in much detail. In particular, the functional viewpoint is nothing more than a placeholder, for example, use cases (one of them provided as an annex to the document), while the deployment viewpoint just skims the surface, introducing the concept of multi-tier systems. With respect to this, however, a very interesting example is made, which shows how the OpenFog approach to deployment is close to IIRA's Layered Databus pattern: it is a hierarchy of layers where nodes on the same level can interact with each other – in what is called "east–west communication" – without the mediation of higher-level entities. The layers themselves, although more relevant to a smart city context, are quite consistent with the IIRA ones. The means by which P2P communication should be implemented are not specified (no databus, in this case).

Besides viewpoints, two additional kinds of frames are used to organize concepts: *views* and *perspectives*. The former include aspects (i.e., *node*, *system* and *software*) that have a clear positioning in the structure of a system, and are further articulated into sub-aspects (e.g., the node view includes security, management, network, accelerators, compute, storage, protocol abstraction and sensors/actuators); the latter are crosscutting concerns (e.g., performance, security, etc.).

Overall, the OpenFog RA gives the impression of being an ambitious exercise, having the main goal of creating a universal conceptual framework that is at the same time generic, comprehensive and detailed. The mapping of a large scale, complex and critical use case (airport visual security), as provided in the document, is impressive, but this comes as no surprise because that was obviously the case study on which the RA itself was fine-tuned. The reverse path – designing a new system using OpenFog RA as the blueprint –

---

[7] The term conveys the concept of cloud computing moved at the ground level

[8] https://www.openfogconsortium.org/ra/

appears to be a daunting task, in particular in industrial scenarios where a very pragmatic approach is the norm. In FAR-EDGE, the value that we see in OpenFog RA is – again, as it was also introduced in IIRA – the concept of a hierarchy of geo-scoped layers that use P2P communication internally.

3.4 FAR-EDGE Reference Architecture

The FAR-EDGE Reference Architecture is the conceptual framework that has driven the design and the implementation of the FAR-EDGE Platform. As an RA, its first goal is communication: providing a terse representation of concepts, roles, structure and behaviour of the system under analysis both internally for the benefit of team members and externally for the sake of dissemination and ecosystem-building. There is a second goal, too, which is reuse: exploiting best practices and lessons learned in similar contexts by the global community of system architects.

The FAR-EDGE RA is described from two architectural viewpoints: the *functional viewpoint* and the *structural viewpoint*. In the sections that follow, they are described in detail. A partial *implementation viewpoint* is also provided further on, with its scope limited to the Ledger Tier. Figure 3.1 provides an overall architecture representation that includes all elements.

3.4.1 Functional Viewpoint

According to the FAR-EDGE RA, the functionality of a factory automation platform can be decomposed into three high-level *Functional Domains* – *Automation*, *Analytics* and *Simulation* – and four *Crosscutting*

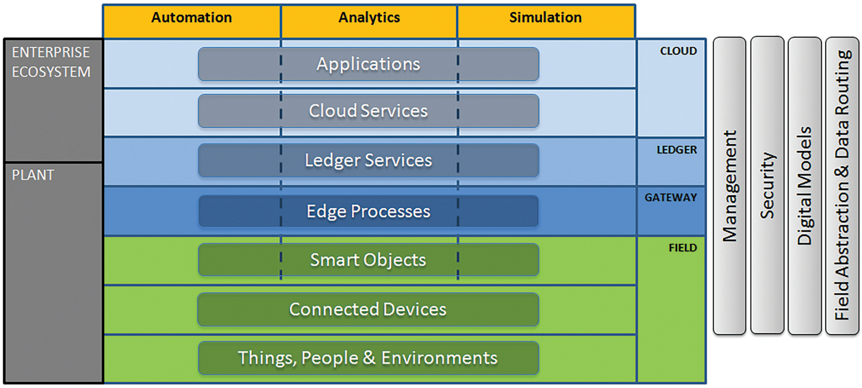

Figure 3.1 FAR-EDGE RA overall view.

(XC) Functions – Management, *Security*, *Digital Models* and *Field Abstraction & Data Routing*. To better clarify the scope of such topics, we have tried to map them to similar IIRA concepts. However, the reader should be aware that the overall scope of the IIRA is wider, as it aims at modelling entire Industrial Internet Systems, while the FAR-EDGE RA is more focused and detailed: oftentimes, concept mapping is partial or even impossible.

Functional Domains and XC Functions are orthogonal to structural Tiers (see next section): the implementation of a given functionality may – but is not required to – span multiple Tiers, so that in the overall architecture representation (Figure 3.1), Functional Domains appear as vertical lanes drawn across horizontal layers. In Figure 3.2 the relationship between Functional Domains, their users and the factory environment is highlighted by arrows showing the flow of data and of control.

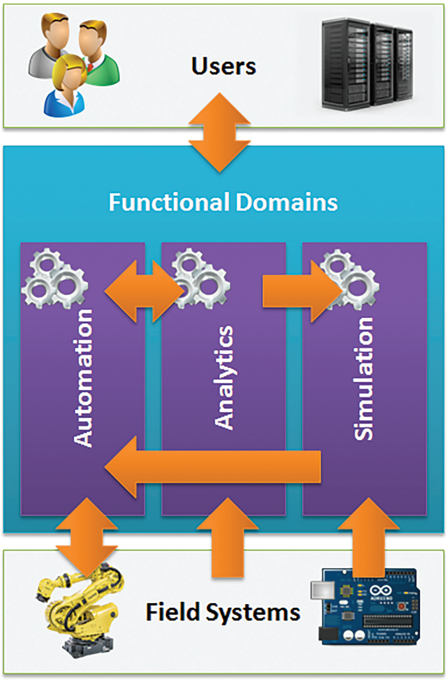

Figure 3.2 FAR-EDGE RA Functional Domains.

3.4.1.1 Automation domain

The FAR-EDGE Automation domain includes functionalities supporting automated control and automated configuration of physical production processes. While the meaning of "control" in this context is straightforward, "configuration" is worth a few additional words. Automated configuration is the enabler of plug-and-play factory equipment – better known as *plug-and-produce* – which in turn is a key technology for mass-customization, as it allows a faster and less expensive adjustment of the production process to cope with a very dynamic market demand. The Automation domain requires a bidirectional monitoring/control communication channel with the Field, typically with low bandwidth but very strict timing requirements (tight control loop). In some advanced scenarios, Automation is controlled – to some extent – by the results of Analytics and/or Simulation (see below for more details on this topic).

The Automation domain partially maps to the Control domain of the IIRA. The main difference is that Control is also responsible for decoupling the real word from the digital world, as it includes the functionality for Field communication, entity abstraction, modelling and asset management. In other words, Control mediates all Field access from other domains like Information, Operations, etc. In the FAR-EDGE RA, instead, the Automation domain is only focused on its main role, while auxiliary concerns are dealt with by Data Models and by Field Abstraction & Data Routing, which are XC Functions.

3.4.1.2 Analytics domain

The FAR-EDGE Analytics domain includes functionalities for gathering and processing Field data for a better understanding of production processes, i.e., a factory-focused business intelligence. This typically requires a high-bandwidth Field communication channel, as the volume of information that needs to be transferred in a given time unit may be substantial. On the other hand, channel latency tends to be less critical than in the Automation scenario. The Analytics domain provides intelligence to its users, but these are not necessarily limited to humans or vertical applications (e.g., a predictive maintenance solution): the Automation and Simulation domains, if properly configured, can both make direct use of the outcome of data analysis algorithms. In the case of Automation, the behaviour of a workflow might change in response to changes detected in the controlled process, e.g., a process drift caused by the progressive wear of machinery or by the quality of assembly components being lower than usual. In the case of Simulation, data analysis can be used to update the parameters of a digital model (see the following section).

The Analytics domain matches perfectly the Information domain of the IIRA, except that the latter is receiving data from the Field through the mediation of Control functionalities.

3.4.1.3 Simulation domain

The FAR-EDGE Simulation domain includes functionalities for simulating the behaviour of physical production processes for the purpose of optimization or of testing what/if scenarios at minimal cost and risk and without any impact of regular shop activities. Simulation requires digital models of plants and processes to be in-sync with the real-world objects they represent. As the real world is subject to change, models should reflect those changes. For instance, the model of a machine assumes a given value of electric power/energy consumption, but the actual values will diverge as the real machine wears down. To detect this gap and correct the model accordingly, raw data from the Field (direct) or complex analysis algorithms (from Analytics) can be used. However, it is important to point out that model synchronization functionality is *not* part of the Simulation domain, which acts just as a consumer of the Digital Models XC Functions.

There is no mapping between the Simulation domain and any functional domain of the IIRA: in the latter, simulation support is not considered as an integral part of the infrastructure.

3.4.1.4 Crosscutting functions

Crosscutting Functions address, as the name suggests, common specific concerns. Their implementation tends to be pervasive, affecting several Functional Domains and Tiers. They are briefly listed and described here.

- **Management:** Low-level functions for monitoring and commissioning/decommissioning of individual system modules, i.e., factory equipment and IT components that expose a management interface. They partially correspond to IIRA's Operations functional domain, with the exclusion of its more high-level functions like diagnostics, prognostics and optimization.
- **Security:** Functions securing the system against the unruly behaviour of its user and of connected systems. These include digital identity

management and authentication, access control policy management and enforcement, communication and data encryption. They partially correspond to the Trustworthiness subset of System Characteristics from IIRA.

- **Digital Models:** Functions for the management of digital models and their synchronization with the real-world entities they represent. Digital modes are a shared asset, as they may be used as the basis for automated configuration, simulation and field abstraction, e.g., semantic interoperability of heterogeneous field systems. They correspond to the Modeling and Asset Management layers of IIRA's Control functional domain.
- **Field Abstraction & Data Routing:** Functions that ensure the connectivity of business logic (FAR-EDGE RA Functional Domains) to the Field, abstracting away the technical details, like device discovery and communication protocols. Data routing refers to the capability of establishing direct producer–consumer channels on demand, optimized for unidirectional massive data streaming, e.g., for feeding Analytics. They correspond to the Communication and Entity Abstraction layers of IIRA's Control functional domain.

3.4.2 Structural Viewpoint

The FAR-EDGE RA uses two classes of concepts for describing the structure of a system: *Scopes* and *Tiers*.

Scopes are very simple and straightforward: they define a coarse mapping of system elements to either the factory – *Plant Scope* – or the broader world of corporate IT – *Enterprise Ecosystem Scope*. Examples of elements in Plant Scope are machinery, field devices, workstations, SCADA and MES systems, and any software running in the factory data centre. To the Enterprise Ecosystem Scope belong ERP and PLM systems and any application or service shared across multiple factories or even companies, e.g., supply chain members.

Tiers are a more detailed and technically oriented classification of deployment concerns: they can be easily mapped to scopes, but they provide more insight into the relationship between system components. Not surprisingly, FAR-EDGE being inspired by edge and distributed computing paradigms, this kind of classification is quite similar to the OpenFog RA's deployment viewpoint, except for the fact that FAR-EDGE Tiers are industry-oriented whereas OpenFog ones are not. That said, FAR-EDGE Tiers are one of the most innovative traits of its RA, and they are individually described here.

3.4.2.1 Field Tier

The Field Tier (see Figure 3.3) is the bottom layer of the FAR-EDGE RA and is populated by *Edge Nodes (EN)*: any kind of device that is connected to the *digital world* on one side and to the *real world* to the other. ENs can have embedded intelligence (e.g., a smart machine) or not (e.g., an IoT sensor or actuator); the FAR-EDGE RA honours this difference: *Smart Objects* are ENs with on-board computing capabilities, and *Connected Devices* are those without. The Smart Object is where local control logic runs: it is a semi-autonomous entity that does not need to interact too frequently with the upper layers of the system.

The Field is also populated by entities of the real world, i.e., those physical elements of production processes that are not directly connected to the network, and as such are not considered as ENs: *Things*, *People* and *Environments*. These are represented in the digital world by some kind of EN "wrapper". For instance, room temperature (Environment) is measured by an IoT sensor (Connected Device), the proximity of a worker (People) to a physical checkpoint location is published by an RFID wearable and detected by an RFID Gate (Connected Device) and a conveyor belt (Thing) is operated by a PLC (Smart Object).

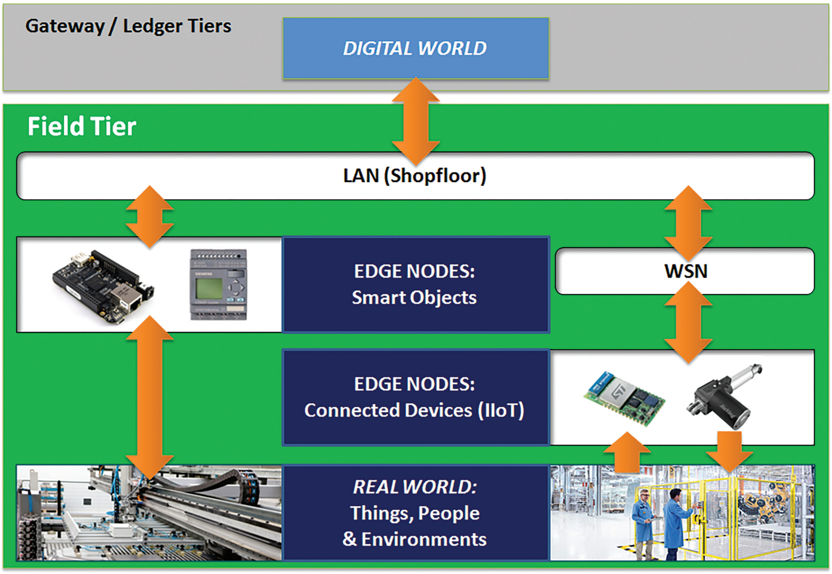

Figure 3.3 FAR-EDGE RA Field Tier.

The Field Tier is in Plant Scope. Individual ENs are connected to the digital world in the upper Tiers either directly by means of the shopfloor's LAN, or indirectly through some special-purpose local network (e.g., WSN) that is bridged to the former.

> From the RAMI 4.0 perspective, the FAR-EDGE Field Tier corresponds to the **Field Device** and **Control Device** levels on the **Hierarchy** axis (IEC-62264/IEC-61512), while the entities there contained are positioned across the **Asset** and **Integration Layers**.

3.4.2.2 Gateway Tier

The Gateway Tier (see Figure 3.4) is the core of the FAR-EDGE RA. It hosts those parts of Functional Domains and XC Functions that can leverage the edge computing model, i.e., software designed to run on multiple, distributed computing nodes placed close to the field, which may include resource-constrained nodes. The Gateway Tier is populated by *Edge Gateways (EG)*: computing devices that act as a digital world gateway to the real world of the Field. These machines are typically more powerful than the average intelligent EN (e.g., blade servers) and are connected to a fast LAN. Strategically positioned close to physical systems, the EG can execute *Edge*

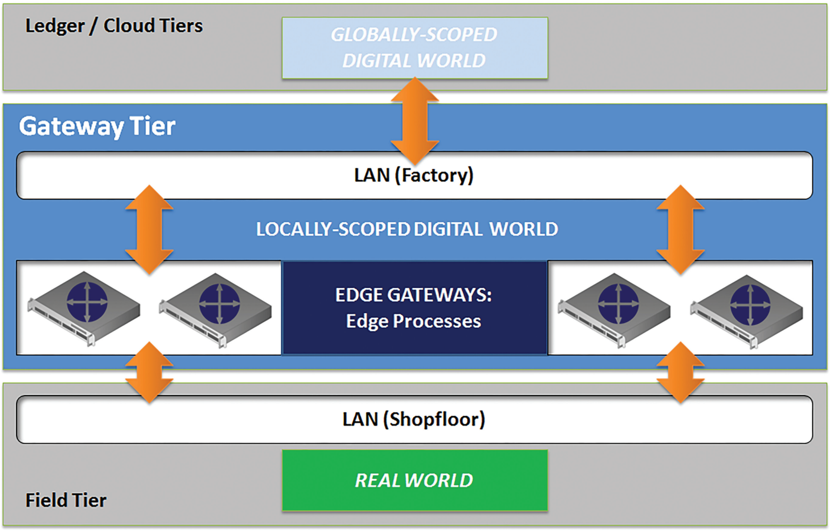

Figure 3.4 FAR-EDGE RA Gateway Tier.

Processes: time- and bandwidth-critical functionality having *local scope*. For instance, the orchestration of a complex physical process that is monitored and operated by a number of sensors, actuators (Connected Devices) and embedded controllers (Smart Objects); or the real-time analysis of a huge volume of live data that is streamed from a nearby Field source.

By itself, the Gateway Tier does not introduce anything new: deploying computing power and data storage in close proximity to where it is actually used is a standard best practice in the industry, which helps reduce network latency and traffic. However, this technique basically requires that the scope of individual subsystems is narrow (e.g., a single work station). If instead the critical functionality applies to a wider scenario (e.g., an entire plant or enterprise), it must be either deployed at a higher level (e.g., the Cloud) – thus losing all benefits of proximity – or run as multiple parallel instances, each focused on its own narrow scope. In the latter case, new problems may arise: keeping global variables in-sync across all local instances of a given process, reaching a consensus among local instances on a “common truth”, collecting aggregated results from independent copies of a data analytics algorithm, etc. These problems are well known: the need for peer nodes of a distributed system to mutually exchange information is recognized by the OpenFog RA. The innovative approach in FAR-EDGE is to define a specific system layer – the Ledger Tier – that is responsible for the implementation of such mechanisms and to guarantee an appropriate Quality of Service level.

The Gateway Tier is in Plant Scope, located above the Field Tier and below the Cloud Tier – in this context, we do not consider the Ledge Tier as part of the north-south continuum, due to its very specific role of support layer. Individual EGs are connected with each other and with the north side of the system – i.e., the globally scoped digital world in the Cloud Tier – by means of the factory LAN, and to the south side through the shopfloor LAN.

From the RAMI 4.0 perspective, the FAR-EDGE Gateway Tier corresponds to the **Station** and **Work Centre** levels on the **Hierarchy** axis (IEC-62264/IEC-61512), while the EGs there contained are positioned across the **Asset**, **Integration** and **Communication Layers**. Edge Processes running on EGs, however, map to the **Information** and **Functional Layers**.

3.4.2.3 Ledger Tier

The Ledger Tier (see Figure 3.5) is a complete abstraction: it does not correspond to any physical deployment environment, and even the entities

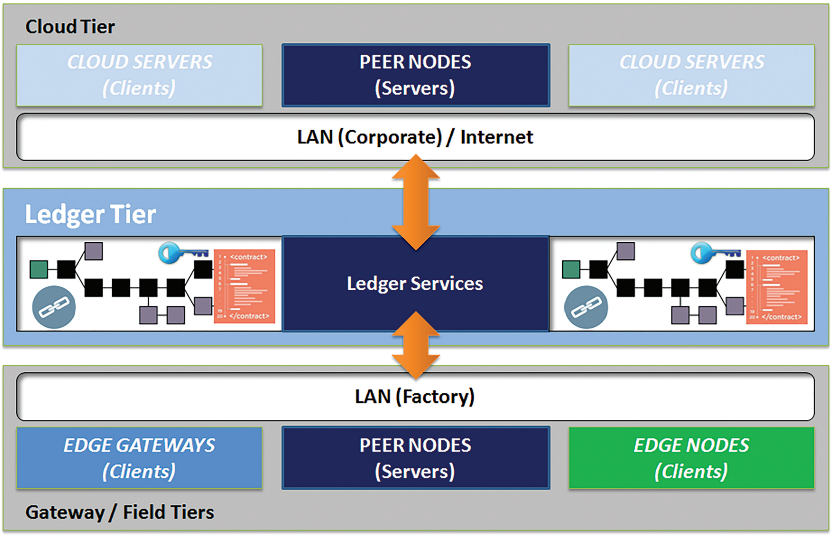

Figure 3.5 FAR-EDGE RA Ledger Tier.

that it “contains” are conventional abstractions. Such entities are *Ledger Services*, which implement decentralized business logic as *smart contracts* executed on a Blockchain platform (see next section for an in-depth technical analysis).

Ledger Services are transaction-oriented: each service call that needs to modify the shared state of a system must be evaluated and approved by *Peer Nodes* before taking effect. Similarly to “regular” services, Ledger Services are implemented as executable code; however, they are not actually executed on any specific computing node: each service call is executed in parallel by all Peer Nodes that happen to be online at the moment, which then need to reach a consensus on its validity. Most importantly, even the executable code of Ledger Services can be deployed and updated online by means of a distributed ledger transaction, just like any other state change.

Ledger Services implement the part of Functional Domains and/or XC Functions that enable the edge computing model, through providing support for their Edge Service counterpart. For example, the Analytics Functional Domain may define a local analytics function (Edge Service) that must be executed in parallel on several EGs, and also a corresponding service call (Ledger Service) that will be invoked from the former each time new or updated local results become available, so that all results can converge into an aggregated dataset. In this case, aggregation logic is included in the

Ledger Service. Another use case may come from the Automation Functional Domain, demonstrating how the Ledger Tier can also be leveraged from the Field: a smart machine with embedded *plug-and-produce* functionality (Smart Object) can ask permission to join the system by making a service call and then, having received green light, can dynamically deploy its own specific Ledger Service for publishing its current state and/or receiving external high-level commands.

The Ledger Tier lays across the Plant and the Enterprise Ecosystem Scopes, as it can provide support to any Tier. The physical location of Peer Nodes, which implement smart contracts and the distributed ledger, is not defined by the FAR-EDGE RA as it depends on implementation choices. For instance, some implementations may use EGs and even some of the more capable ENs in the role of Peer Nodes; others may separate concerns, relying on specialized computing nodes that are deployed on the Cloud.

From the RAMI 4.0 perspective, the FAR-EDGE Ledger Tier corresponds to the **Work Centre**, **Enterprise** and **Connected World** levels on the **Hierarchy** axis (IEC-62264/IEC-61512), while the Ledger Services there contained are positioned across the **Information** and **Functional Layers**.

3.4.2.4 Cloud Tier

The Cloud Tier (see Figure 3.6) is the top layer of the FAR-EDGE RA, and also the simplest and more "traditional" one. It is populated by *Cloud Servers (CS)*: powerful computing machines, sometimes configured as clusters, that are connected to a fast LAN internally to their hosting data centre, and made accessible from the outside world by means of a corporate LAN or the Internet. On CSs runs that part of the business logic of Functional Domains and XC Functions that benefits from having the widest of scopes over production processes, and can deal with the downside of being physically deployed far away from them. This includes the planning, monitoring and management of entire factories, enterprises and supply chains (e.g., MES, ERP and SCM systems). The Cloud Tier is populated by *Cloud Services* and *Applications*. The difference between them is straightforward: Cloud Services implement specialized functions that are provided as individual API calls to Applications, which instead "package" a wider set of related operations that are relevant to some higher-level goal and often – but not necessarily – expose an interactive human interface.

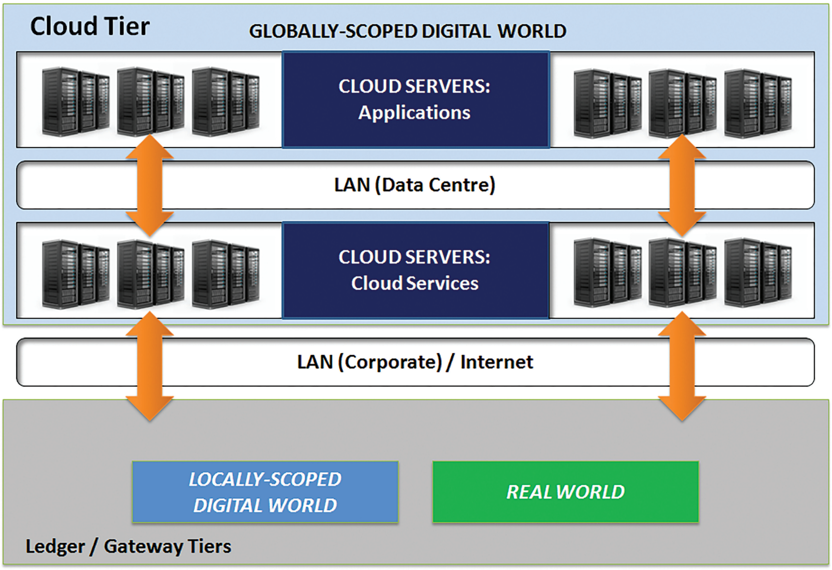

Figure 3.6 FAR-EDGE Cloud Tier.

The Cloud Tier is in Enterprise Ecosystem scope. The "Cloud" term in this context implies that Cloud Services and Applications are visible from all Tiers, wherever located. It does *not* imply that CSs should be actually hosted on some commercial ISP's infrastructure. More often, in particular in large enterprises, the Cloud Tier corresponds to one or more corporate data centres (private cloud), ensuring that the entire system is fully under the control of its owner.

From the RAMI 4.0 perspective, the FAR-EDGE Cloud Tier corresponds to the **Work Centre**, **Enterprise** and **Connected World** levels on the **Hierarchy** axis (IEC-62264/IEC-61512), while the Cloud Services and Applications there contained are positioned across the **Information**, **Functional** and **Business Layers**.

3.5 Key Enabling Technologies for Decentralization

In this section, our main concern is the use of Blockchain and smart contracts as the key enabling technologies of Ledger Services (see the Ledger Tier section above). In FAR-EDGE, the baseline Blockchain platform is an off-the-shelf product, which is enriched by application-specific smart contract software. That said, there are some Blockchain-related basic issues that we need to account for.

3.5.1 Blockchain Issues

For those familiar with the technology, the main question is: how can a Blockchain fit industrial automation scenarios? According to conventional wisdom, Blockchains are slow and cumbersome systems with limited scalability and an aversion to data-intensive applications. Nevertheless, while this vision has solid roots in reality, in the context of smart factories, these shortcomings are not as relevant as it may seem. In order to substantiate this claim, though, we first need to explain some key points of the technology.

First and foremost, the Blockchain is a log of all transactions (i.e., state changes) executed in the system. The log, which is basically a witness of past and current system states, is replicated and kept in-sync across multiple nodes. All nodes are peers, so that no “master node” or “master copy” of the log exists anywhere at any time. Internally, the log is a linear sequence of records (i.e., *blocks* containing transactions) that are individually immutable and time-stamped. The sequence itself can only be modified by appending new records at the end. The integrity of both records and sequence is protected by means of strong cryptographic algorithms [3]. Moreover, all records must be approved by consensus among peers, using some sort of *Byzantine Fault Tolerance (BFT)* mechanism as a guarantee that an agreement on effective system state can always be reached, even if some peers are unavailable or misbehaving (in good faith or for malicious purposes) [4, 5].

The process described above is all about trust: the consensus protocol guarantees that all approved transactions conform to the business logic that peers have agreed on, while the log provides irrefutable evidence of transactions. For this to work in a zero-trust environment, where peers that do not know (let alone trust) each other and are not subject to a higher authority, there is yet another mechanism in place: an economic incentive that rewards “proper” behaviour and makes the cost of cheating much higher than the profit. Given that the whole system must be self-contained and autonomous, such incentive is based on native digital money: a *cryptocurrency*. This closes the loop: all public Blockchain networks need *cryptoeconomics* to make their BFT mechanism work. For some of them (e.g., Bitcoin), the cryptocurrency itself is the main goal of the system: transactions are only used to exchange value between users. Other systems (e.g., Ethereum) are much more flexible, as we will see further on. That said, cryptocurrencies are problematic for many reasons, including regulatory compliance, and hinder the adoption of the Blockchain in the corporate world.

Another key point of Blockchain technology that is worth mentioning is the problem of *transaction finality*. Most BFT implementations rely on *forks* to resolve conflicts between peer nodes: when two incompatible opinions on the validity of some transaction exist, the log is split into two branches, each corresponding to one alternate vision of reality, i.e., of system state. The other nodes of the network will then have to choose which branch is the valid one, and will do this by appending their new blocks to the "right" branch only. Over time, consensus will coalesce on one branch (the one having more new blocks appended), and the losing branch will be abandoned. While this scheme is indeed effective for achieving BFT in public networks, it has one important consequence: there is no absolute guarantee that a committed transaction will stay so, because it may be deemed invalid *after* it is written to the log. In other words, it may appear only on the "bad" branch of a fork and be reverted when the conflict is resolved. Clearly enough, this behaviour of the Blockchain is not acceptable in scenarios where a committed transaction has side effects on other systems.

This is how first-generation Blockchains work. For all these reasons, public Blockchains are, at least to date, extremely inefficient for common online transaction processing (OLTP) tasks. This is most unfortunate, because second-generation platforms like Ethereum have introduced the smart contract concept. Smart contracts were initially conceived as a way for users to define their custom business logic for transaction, i.e., making the Blockchain "smarter" by extending or even replacing the built-in logic of the platform. It then became clear that smart contracts, if properly leveraged, could also turn a Blockchain into a distributed computing platform with unlimited potential. However, distributed applications would still have to deal with the scalability, responsiveness and transaction finality of the underlying BFT engine, which significantly limits the range of possible use cases.

To tackle this problem, the developer community is currently treading two separate paths: upgrading the BFT architecture on the one hand and relax functional requirements on the other hand. The former approach is ambitious but slow and difficult: it is followed by a third generation of Blockchain platforms that are proposing some innovative solution, although transaction finality still appears to be an open point nearly everywhere. The latter is much easier: if we can assume some limited degree of trust between parties, we can radically simplify the BFT architecture and thus remove the worst bottlenecks. From this reasoning, an entirely new species was born in recent years: *permissioned* Blockchains. Given their simpler architecture, commercial-grade permissioned Blockchains are already available today (e.g., Hyperledger, Corda), as opposed to third-generation ones (e.g., EOS, NEO) which are still experimental.

3.5.2 Permissioned Blockchains

Permissioned Blockchains are second-generation architectures that do not support anonymous nodes and do not rely on cryptoeconomics. Basically, they are meant to make the power of Blockchain and smart contracts available to the enterprise, at least to some extent. Their BFT is still a decentralized process executed by peer nodes; however, the process runs under the supervision of a central authority. This means that all nodes must have a strong digital identity (no anonymous parties) and be trusted by the authority in order to join the system. Trust, and thus access to the Blockchain, can be revoked at any time. The BFT protocol can then rely on some basic assumptions and perform much faster, narrowing the distance from OLTP standards in terms of both responsiveness and throughput. Some BFT implementation also support final transactions, as consensus on transaction validity can be reached in near-real-time *before* anything is written to the log.

The key point of permissioned Blockchains is that they are only partially decentralized, leaving governance and administration roles in the hands of a leading entity – be it a single organization or a consortium. This aspect is a boon for enterprise adoption, for obvious reasons. Typically, these networks are also much smaller than public ones, with the positive side effect of limiting the inefficiency of data storage caused by massive data replication across peer nodes. Overall, we can argue that permissioned Blockchains are a viable compromise between the original concept and legacy OLTP systems. But then, to what extent? Can we identify some use cases that a state-of-the-art permissioned Blockchain can effectively support? This is exactly what the FAR-EDGE project aims at, with the added goal of validating claims on the field, by means of pilot applications deployed in real-world industrial environments.

3.5.3 The FAR-EDGE Ledger Tier

The first problem that FAR-EDGE had to face was to define the *performance envelope* of current Blockchain implementations, so that validation cases could be shaped according to the sustainable workload. The idea was to set the benchmark for a Blockchain *comfort zone* in terms of a few objective and measurable Key Performance Indicators (KPI), targeting the known weak points of the technology:

- *Transaction Average Latency (TrxAL)* – The average waiting time for a client to get confirmation of a transaction, expressed in seconds.
- *Transaction Maximum Sustained Throughput (TrxMST)* – The maximum number of transactions that can be processed in a second, on average.

The benchmark was set by stress-testing, in a lab environment, actual Blockchain platforms. These were selected after a preliminary analysis of the permissioned Blockchains available from open source communities, using criteria like code maturity and, most importantly, finality of transactions. The only two platforms that passed the selection were Hyperledger Fabric (HLF) and NEO. The stress test was then conducted using BlockBench, a specialized testing framework [6], and a simple configuration of eight nodes on commodity hardware.

HLF emerged from tests as the only viable platform for CPS applications, given that NEO is penalized by a significant latency ($\sim$7s.), which is independent of workload (the expected result for a "classical" Blockchain architecture that aggregates transactions into blocks and defines a fixed delay for processing each block). On the contrary, HLF was able to accept a workload of up to 160 transactions per second with relatively low latency (0.1–1 s.). On heavier workloads, up to 1000 transactions per second, NEO is instead the clear winner, thanks to its constant latency, while HLF's performance progressively degrades (>50 s.). This workload profile however, while appealing for high-throughput scenarios (e.g., B2C payment networks), is not compatible with basic CPS requirements. Consequently, the Blockchain performance benchmark was set as follows:

- <= TrxAL <= 1.0
- 0 <= TrxMST <= 160

This is also considered the performance envelope of the FAR-EDGE Ledger Tier, as the HLF platform has been adopted as its baseline Blockchain implementation.

3.5.4 Validation use Cases

Having marked some boundaries, the FAR-EDGE project then proceeded with the identification of some pilot applications for the validation phase. The starting point was a set of candidate use cases proposed by our potential users, who were eager to tackle some concrete problems and experiment with some new ideas. The general framework of this exercise is described here.

As explained, the main objective in FAR-EDGE is to achieve flexibility in the factory through the decentralization of production systems. The catalyst of this transformation is the Blockchain, which – if used as a computing platform rather than a distributed ledger – allows the virtualization of the automation pyramid. The Blockchain provides a common *virtual space* where data can be securely shared and business logic can be consistently run. That said, users can leverage this opportunity in two ways: one easier but somewhat limited approach, and the other more difficult and more ambitious approach.

The easiest approach is of the brown-field type: just migrate (some of) the factory's centralized monitoring and control functionality to Ledger Services on the Ledger Tier. Thanks to the Gateway Tier, legacy centralized services can be "impersonated" on a local scale by Edge Gateways: the shopfloor – that hardest environment to tamper with in a production facility – is left untouched. The main advantages of this configuration are the mitigation of performance bottlenecks (heavy network traffic is confined locally, workload is spread across multiple computing nodes) and added resiliency (segments of the shopfloor can still be functional when temporarily disconnected from the main network). Flexibility is also enhanced, but on a coarse-grained scale, modularity is achieved by grouping a number of shopfloor Edge Nodes under the umbrella of one Edge Gateway, so that they all together become a single "module" with some degree of self-contained intelligence and autonomy. Advanced Industry 4.0 scenarios like plug-and-produce are out of reach.

The more ambitious approach is also a much more difficult and risky endeavour in real-world business, being of the green-field type. It is about delegating responsibility to Smart Objects on the shopfloor, which communicate with each other through the mediation of the Ledger Tier. The business logic in Ledger Services is of higher level with respect to the previous scenario: more about governance and orchestration than direct control. The Gateway Tier has a marginal role, mostly confined to Big Data analytics. In this configuration, central bottlenecks are totally removed and the degree of flexibility is extreme. The price to pay is that a complete overhaul of the shopfloor of existing factories is required, replacing PLC-based automation with intelligent machines.

In FAR-EDGE, both paths are explored with different use cases combining on automation, analytics and simulation. We here give one full example of each type.

The first use case follows the brown-field approach. The legacy environment is an assembly facility for industrial vehicles. The pilot is called *mass-customization*: the name refers to capability of the factory assembly line to handle individually customized products having a high level of variety. If implemented successfully, mass-customization can give a strategic advantage to target niche markets and meet diverse customer needs in a timely fashion. In particular, the pilot factory produces highly customized trucks. The product specification is defined by up to 800 unique variants, and the final assembly includes approximately 7000 manufacturing operations and handles a very high degree of geometrical variety (axle configurations, fuel tank positions etc.). Despite the high level of variety in the standard product, at some production sites, 60% of the produced trucks have unique customer adaption.

In the pilot factory, the main assembly line is sequential but feeds a number of finishing lines that work in parallel. In particular, the wheel alignment verification is done on the finishing assembly line and is one of the last active checks done on trucks before they leave the plant. This opens up an opportunity to optimize the workload. In the as-is scenario, wheel alignment stations are statically configured to accommodate specific truck model ranges: products must be routed to a matching station on arrival, creating a potential bottleneck if model variety is not optimal. As part of the configuration, a handheld nut runner tool needs to be instructed as to the torque force to apply.

In the to-be solution, according to the FAR-EDGE architectural blueprint, each wheel alignment station is represented at the Gateway Tier level by a dedicated Edge Gateway box. The EG runs some simple ad-hoc automation software that integrates the Field systems attached to the station (e.g., a barcode reader, the smart nut runner) using standard IoT protocols like MQTT. The EG also runs a peer node that is a member of the logical Ledger Tier. A custom Ledger Service deployed on the Ledger Tier implements the business logic of the use case. The instruction set for the products to be processed is sent in JSON format to the Ledger Service, once per day, by the central ERP-MES systems: from that point and until a new production plan is published, the Ledger and Gateway Tiers are autonomous.

When a new truck reaches the end of the main line, it is dispatched to the first finishing line available, achieving the desired result of product flow optimization. Then, when it reaches the *wheel alignment station*, the chassis ID is scanned by a barcode reader and a request for instructions is sent, through the automation layer on the EG, to the Ledger Service. The Ledger Service will retrieve the instruction set from the production

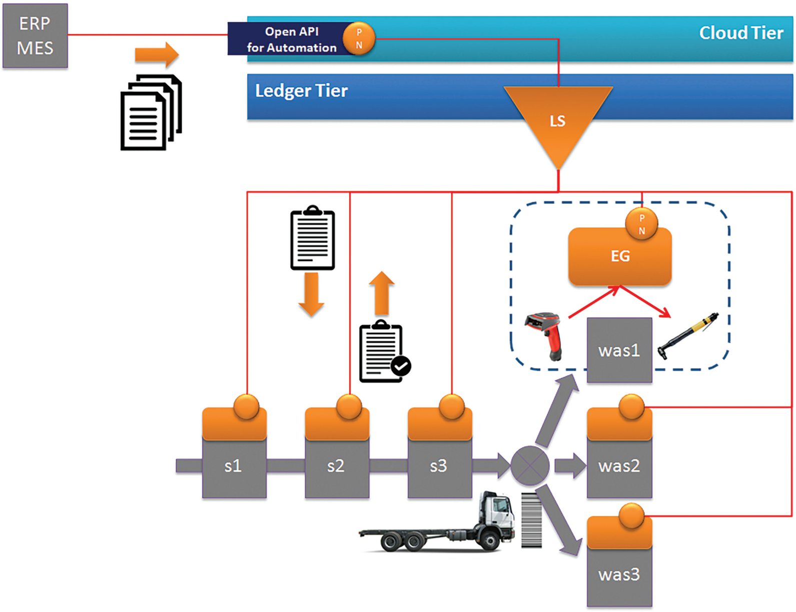

Figure 3.7 Mass-customization use case.

plan – which is saved on the Ledger itself – by matching the chassis ID. When the automation layer receives the instructions set, it parses the specific configuration parameters of interest and sends them to the nut runner, which adjusts itself. The wheel alignment operations will then proceed as usual. A record of the actual operations performed, which may differ from those in the instruction set, is finally set back to the Ledger and used to update the production plan. An overall view of the use case is given in Figure 3.7.

While the product flow optimization mentioned above is the immediate result of the pilot, there are some additional benefits to be gained either as a by-product or as planned extensions.

First, the wheel alignment station, together with its EG box, becomes an autonomous module that can be easily added/removed and even relocated in a different environment. This scenario is not as far-fetched as it may seem, because it actually comes from a business requirement: the company has a number of production sites in different locations all over the world, each with their own unique MES maps. The deployment of a new module with different MES maps is currently a difficult and costly process.

Second, in the future, the truck itself may become a Smart Object that communicates directly with the Ledger Tier. Truck–Ledger interactions will happen throughout the entire life cycle of the truck – from manufacturing to operation and until decommissioning – with the Ledger maintaining a digital twin of the truck.

The second use case follows instead the *heavyweight* green-field approach. The pilot belongs to a white goods (i.e., domestic appliances) factory. The objective of the pilot is “reshoring”, which in the FAR-EDGE context means enabling the company to move production back from off-shore locations, thanks to a better support for the rapid deployment of new technologies (i.e., shopfloor Smart Objects) offered by the more advanced domestic plants. In this particular plant, a 1 km long conveyor belt moves pallets of finished products from the factory to a warehouse, where they are either stocked or forwarded for immediate delivery. The factory/warehouse conveyor is not only a physical boundary, but also an administrative one, as the two facilities are under the responsibility of two different business units. Moreover, once the pallet is loaded on a delivery vehicle, it comes under the responsibility of a third party who operates the delivery business.

In the as-is scenario, the conveyor feeds 19 shipping bays, or “lanes”, in the warehouse. Each lane is simply a dead-end conveyor segment, where pallets are dropped in by the conveyor and retrieved by a manually operated forklift (basically, an FIFO queue). Simple mechanical actuators do the physical routing of the pallets, controlled by logic that runs on a central “sorter” PLC. The sorting logic is very simple: it is based on a production schedule that is defined once per day and on static mappings of the lanes to product types and/or final destinations. This approach has one big problem: production cannot be dynamically tuned to match business changes, or at least only to a very limited extent, because the fixed dispatching scheme downstream cannot sync with it. The problem is not only in software: the physical layout of the system is fixed.

In the to-be solution, the shipping bays become Smart Objects that can be plugged in and out at need (see Figure 3.8). They embed simple sensors that detect the number of pallets currently in their local queue, and a controller board that runs some custom automation logic and connects directly to the Ledger Tier (i.e., without the mediation of an Edge Gateway). A custom Ledger Service acts as a coordination hub: it is responsible for authorizing a new “smart bay” that advertise itself to join the system (plug-and-produce) and, once accepted, to apply the sorting logic. This is based on the current state of the main conveyor belt, where incoming and outgoing pallets are individually identified by an RFID tag, and on “capability update” messages that are sent by smart bays each time they undergo an internal state change

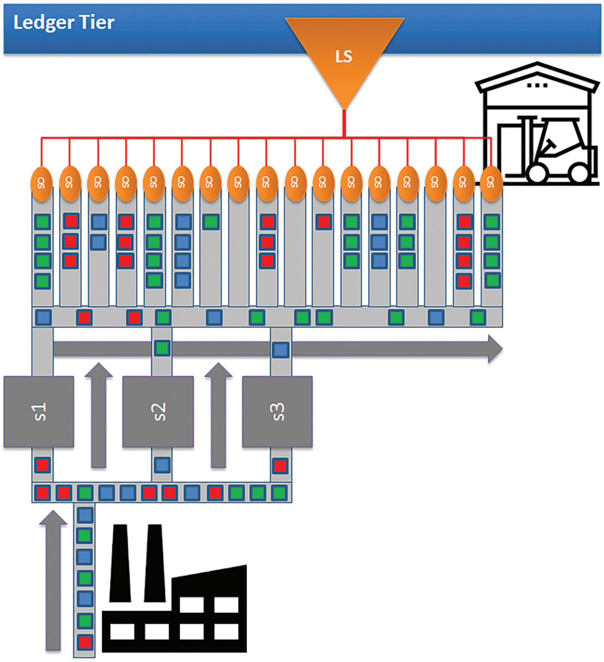

Figure 3.8 Reshoring use case.

(e.g., number of free slots in the local queue, preference for a product type). The production schedule is not required at all, because sorting is only calculated on the actual state.

3.6 Conclusions

FAR-EDGE is one of the few ongoing initiatives that focus on edge computing for factory automation, similarly to the IIC's edge intelligence testbed and EdgeX Foundry. However, the FAR-EDGE RA introduces some unique concepts. In particular, the notion of a special logical layer, the Ledger Tier, that is responsible for sharing process state and enforcing business rules across the computing nodes of a distributed system, thus permitting *virtual* automation and analytics processes that span multiple nodes – or, from a bottom-up perspective, autonomous nodes that cooperate to a common goal. This new kind of architectural layer stems from the availability of Blockchain technology, which, while being well understood and thoroughly tested in mission-critical areas like digital currencies, have never been applied before to industrial systems. FAR-EDGE aims at demonstrating how a pool of specific Ledger Services can enable decentralized factory automation in an effective, reliable, scalable and secure way. In this chapter, we also presented the general framework of the industrial pilot applications that are going to be run during the validation phase of the project.

References

[1] Karsten Schweichhart: Reference Architectural Model Industrie 4.0 – An Introduction, April 2016, Deutsche Telekom, online resource: https://ec.europa.eu/futurium/en/system/files/ged/a2-schweichhart-reference_architectural_model_industrie_4.0_rami_4.0.pdf

[2] Dagmar Dirzus, Gunther Koschnick: Reference Architectural Model Industrie 4.0 – Status Report, July 2015, VDI/ZVEI, online resource: https://www.zvei.org/fileadmin/user_upload/Themen/Industrie_4.0/Das_Referenzarchitekturmodell_RAMI_4.0_und_die_Industrie_4.0-Komponente/pdf/5305_Publikation_GMA_Status_Report_ZVEI_Reference_Architecture_Model.pdf

[3] H. Halpin, M. Piekarska, "Introduction to Security and Privacy on the Blockchain", IEEE European Symposium on Security and Privacy Workshops (EuroS & PW), Paris, 2017, pp. 1–3.

[4] L. Lamport, R. Shostak, M. Pease, "The Byzantine Generals problem", ACM Transactions on Programming Languages and Systems, volume 4 no. 3, p. 382–401, 1982.

[5] Z. Zheng, S. Xie, H. Dai, X. Chen, H. Wang, "An overview of Blockchain technology: architecture, consensus, and future trends", proceedings of IEEE 6th International Congress on Big Data, 2017.

[6] T. Dinh, J. Wang, G. Chen, R. Liu, C. Ooi, K. L. Tan, "BLOCKBENCH: a framework for analyzing private Blockchains", unpublished, 2017. Retrieved from: https://arxiv.org/pdf/1703.04057.pdf