Unit 5 Review

IoT Reference Architecture

- Introduction, Functional View, Information View, Deployment and Operational View, Other Relevant architectural views.

Real-World Design Constraints

- Introduction, Technical Design constraints-hardware is popular again, Data representation and visualization, Interaction and remote control.

Industrial Automation

- Service-oriented architecture-based device integration, SOCRADES: realizing the enterprise integrated Web of Things, IMC-AESOP: from the Web of Things to the Cloud of Things,

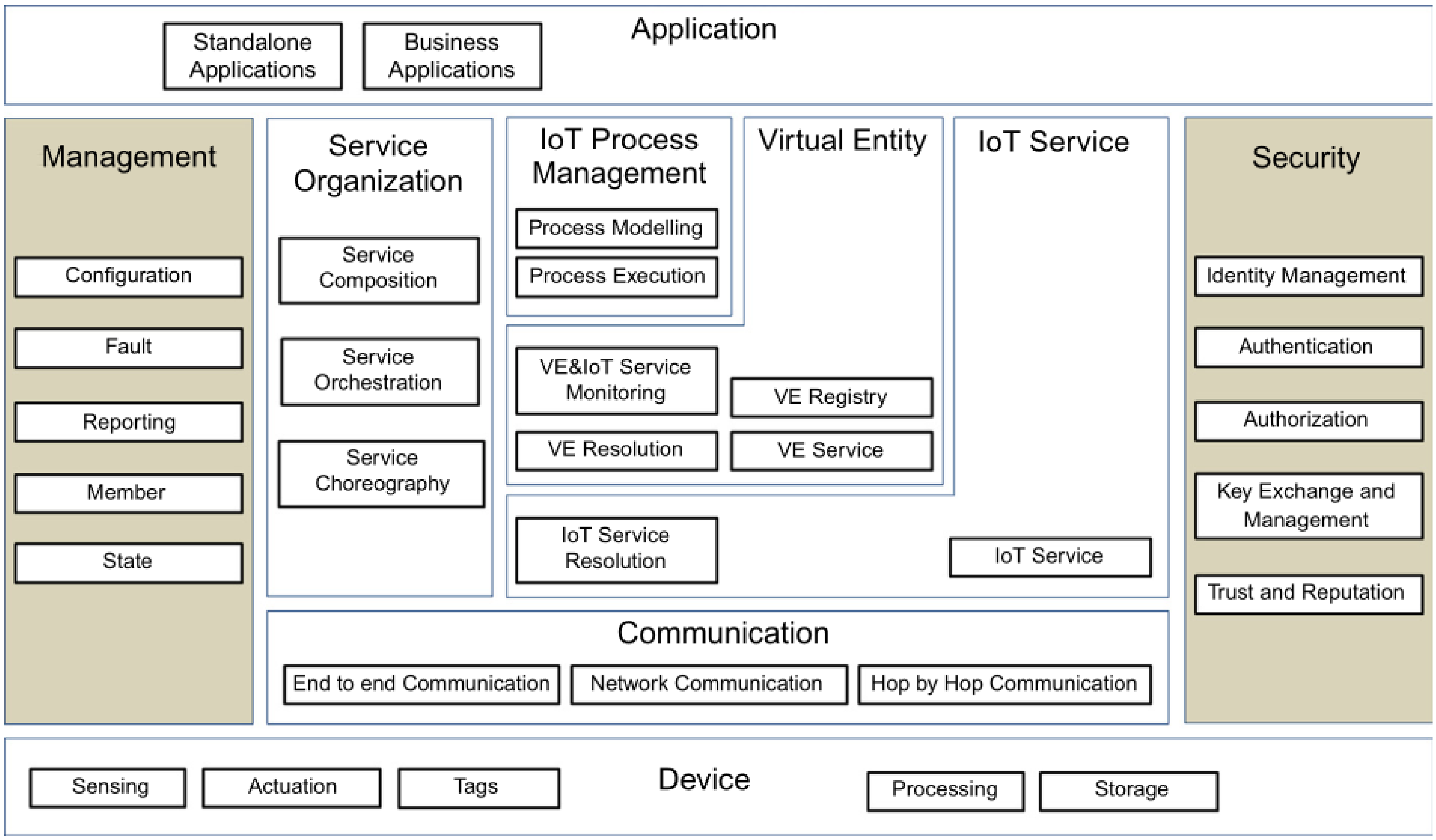
IoT Reference Architecture

- The Reference Architecture is a starting point for generating concrete architectures and actual systems.
- A concrete architecture addresses the concerns of multiple stakeholders of the actual system, and it is typically presented as a series of views that address different stake-holder concerns.
- Views are useful for reducing the complexity of the Reference Architecture blueprints by addressing groups the concerns of one group at a time.

Views

- The stakeholders for a concrete IoT system are the people who use the system (Human Users); the people who design, build, and test the Resources, Services, Active Digital Artifacts, and Applications;
- The people who deploy Devices and attach them to Physical Entities; the people who integrate IoT capabilities of functions with an existing ICT system (e.g. of an enterprise);
- The people who operate, maintain, and troubleshoot the Physical and Virtual Infrastructure;
- The people who buy and own an IoT system or parts thereof
- In order to address the concerns of mainly the concrete IoT architect, and secondly the concerns of most of the above stakeholders, the Reference Architecture is presented as a set of architectural views
- **Functional View:** Description of what the system does, and its main functions.
- **Information View:** Description of the data and information that the system handles.
- **Deployment and Operational View:** Description of the main real world components of the system such as devices, network routers, servers, etc.

IoT Functional View



IoT Functional View.

Device and Application functional group

- Device FG contains the Sensing, Actuation, Tag, Processing, Storage FCs, or simply components.
- These components represent the resources of the device attached to the Physical Entities of interest. The Application FG contains either standalone applications (e.g. for iOS, Android, Windows phone), or Business Applications that connect the IoT system to an Enterprise system.

Communication functional group

- The Communication FG contains the End-to-End Communication, Network Communication, and Hop-by-Hop communication components:
- The Hop-by-Hop Communication is applicable in the case that devices are equipped with mesh radio networking technologies such as IEEE 802.15.4 for which messages have to traverse the mesh from node-to-node (hop-by-hop) until they reach a gateway node which forwards the message (if needed) further to the Internet.

Network FC

- The Network FC is responsible for message routing & forwarding and the necessary translations of various identifiers and addresses.
- The translations can be (a) between network layer identifiers to MAC and/or physical network identifiers,
- (b) between high-level human readable host/node identifiers to network layer addresses (e.g. Fully Qualified Domain Names (FQDN) to IP addresses, a function implemented by a Domain Name System (DNS) server),
- (c) translation between node/service identifiers and network locators in case the higher layers above the networking layer use node or service identifiers that are decoupled from the node addresses in the network (e.g. Host Identity Protocol)

End to End Communication

- The End-to-End Communication FC is responsible for end-to-end transport of application layer messages through diverse network and MAC/PHY layers.
- In turn, this means that it may be responsible for end-to-end retransmissions of missing frames depending on the configuration of the FC.
- For example, if the End-to-End Communication FC is mapped in an actual system to a component implementing the Transmission Control Protocol (TCP) protocol, reliable transfer of frames dictates the retransmission of missing frames

IoT Service functional group - The IoT Service FC

- IoT Service functional group The IoT Service FG consists of two FCs:
- The IoT Service FC and the IoT Service Resolution FC:
- The IoT Service FC is a collection of service implementations, which interface the related and associated Resources.
- For a Sensor type of a Resource, the IoT Service FC includes Services that receive requests from a User and returns the Sensor Resource value in synchronous or asynchronous (e.g. subscription/notification) fashion.

IoT Service functional group - The IoT Service Resolution FC

- The IoT Service Resolution FC contains the necessary functions to realize a directory of IoT Services that allows dynamic management of IoT Service descriptions and discovery/lookup/resolution of IoT Services by other Active Digital Artifacts.
- Dynamic management includes methods such as creation/update/deletion (CUD) of Service description, and can be invoked by both the IoT Services themselves, or functions from the Management FG.
- The discovery/lookup and resolution functions allow other Services or Active Digital Artifacts to locate IoT Services by providing different types of information to the IoT Service Resolution FC.

Virtual Entity functional group

- The Virtual Entity FG contains functions that support the interactions between Users and Physical Things through Virtual Entity services.
- An example of such an interaction is the query to an IoT system of the form, “What is the temperature in the conference room Titan?”
- The Virtual Entity is the conference room “Titan,” and the conference room attribute of interest is “temperature.”
- **The Virtual Entity Service FC** enables the interaction between Users and Virtual Entities by means of reading and writing the Virtual Entity attributes (simple or complex), which can be read or written.
- **The Virtual Entity Registry FC** maintains the Virtual Entities of interest for the specific IoT system and their associations. The component offers services such as creating/reading/updating/deleting Virtual Entity descriptions and associations.

Virtual Entity functional group

- The Virtual Entity Resolution FC maintains the associations between Virtual Entities and IoT Services, and offers services such as creating/reading/updating/deleting associations as well as lookup and discovery of associations.
- The Virtual Entity and IoT Service Monitoring FC includes:
- (a) functionality to assert static Virtual Entity - IoT Service associations,
- (b) functionality to discover new associations based on existing associations or Virtual Entity attributes such as location or proximity, and
- (c) continuous monitoring of the dynamic associations between Virtual Entities and IoT Services and updates of their status in case existing associations are not valid any more.

IoT process management functional group

- The IoT Process Management FG aims at supporting the integration of business processes with IoT-related services.
- It consists of two FCs:
- **The Process Modeling FC** provides that right tools for modeling a business process that utilizes IoT-related services.
- **The Process Execution FC** contains the execution environment of the process models created by the Process Modelling FC and executes the created processes by utilizing the Service Organization FG in order to resolve high-level application requirements to specific IoT services.

Service Organization functional group

- The Service Organization FG acts as a coordinator between different Services offered by the system.

It consists of the following FCs:

- **The Service Composition FC** manages the descriptions and execution environment of complex services consisting of simpler dependent services.
- An example of a complex composed service is a service offering the average of the values coming from a number of simple Sensor Services.
- **The Service Orchestration FC** resolves the requests coming from IoT Process Execution FC or User into the concrete IoT services that fulfil the requirements.
- **The Service Choreography FC** is a broker for facilitating communication among Services using the Publish/Subscribe pattern.

Security functional group

- The Security FG contains the necessary functions for ensuring the security and privacy of an IoT system. It consists of the following FCs:
- The Identity Management FC manages the different identities of the involved Services or Users in an IoT system in order to achieve anonymity.
- The Authentication FC verifies the identity of a User and creates an assertion upon successful verification.
- It also verifies the validity of a given assertion.
- **The Authorization FC** manages and enforces access control policies. It provides services to manage policies (CUD), as well as taking decisions and enforcing them regarding access rights of restricted resources. The term “resource” here is used as a representation of any item in an IoT system that needs a restricted access.
- Such an item can be a database entry (Passive Digital Artifact), a Service interface, a Virtual Entity attribute (simple or complex), a Resource/Service/Virtual Entity description, etc.
- **The Key Exchange & Management** is used for setting up the necessary security keys between two communicating entities in an IoT system. This involves a secure key distribution function between communicating entities.
- **The Trust & Reputation FC** manages reputation scores of different interacting entities in an IoT system and calculates the service trust levels.

Management functional group

- The Management FG contains system-wide management functions that may use individual FC management interfaces. It is not responsible for the management of each component, rather for the management of the system as a whole.

It consists of the following FCs:

- **The Configuration FC** maintains the configuration of the FCs and the Devices in an IoT system (a subset of the ones included in the Functional View).
- The component collects the current configuration of all the FCs and devices, stores it in a historical database, and compares current and historical configurations.
- The component can also set the system-wide configuration (e.g. upon initialization), which in turn translates to configuration changes to individual FCs and devices.
- **The Fault FC** detects, logs, isolates, and corrects system-wide faults if possible. This means that individual component fault reporting triggers fault diagnosis and fault recovery procedures in the Fault FC.
- **The Member FC** manages membership information about the relevant entities in an IoT system. Example relevant entities are the FGs, FCs, Services, Resources, Devices, Users, and Applications. Membership information is typically stored in a database along with other useful information such as capabilities, ownership, and access rules & rights, which are used by the Identity Management and Authorization FCs.
- **The State FC** is similar to the Configuration FC, and collects and logs state information from the current FCs, which can be used for fault diagnosis, performance analysis and prediction, as well as billing purposes. This component can also set the state of the other FCs based on system-wise state information.
- **The Reporting FC** is responsible for producing compressed reports about the system state based on input from FCs.

Information view

- The information view consists of
- (a) the description of the information handled in the IoT System, and
- (b) the way this information is handled in the system; in other words, the information lifecycle and flow (how information is created, processed, and deleted), and the information handling components.

- The pieces of information handled by an IoT system it can be
- Virtual Entity context information, i.e. the attributes (simple or complex) as represented by parts of the IoT Information model.
- IoT Service output itself is another important part of information generated by an IoT system. For example, this is the information generated by interrogating a Sensor or a Tag Service
- Virtual Entity descriptions in general, which contain not only the attributes coming from IoT Devices (e.g. ownership information).
- Associations between Virtual Entities and related IoT Services.

Information flow and lifecycle

- On a high level, the flow of information in an IoT system follows two main directions.
- From devices that produce information such as sensors and tags, information follows a context-enrichment process until it reaches the consumer application or part of the larger system, and from the application or part of a larger system information it follows a context-reduction process until it reaches the consumer types of devices.

Cont..

- The enrichment process is shown in Figure. Devices equipped with sensors transform changes in the physical properties of the Physical Entities of Interest into electrical signals.
- These electrical signals are transformed in one or multiple values (Figure 8.2a) on the device level.
- These values are then enriched with metadata information such as units of measurement, timestamp, and possibly location information (Figure 8.2b).
- These enriched values are offered by a software component (Resource) either on the device or the network. The Resource exposes certain IoT Services to formalize access to this enriched information (Figure 8.2c).

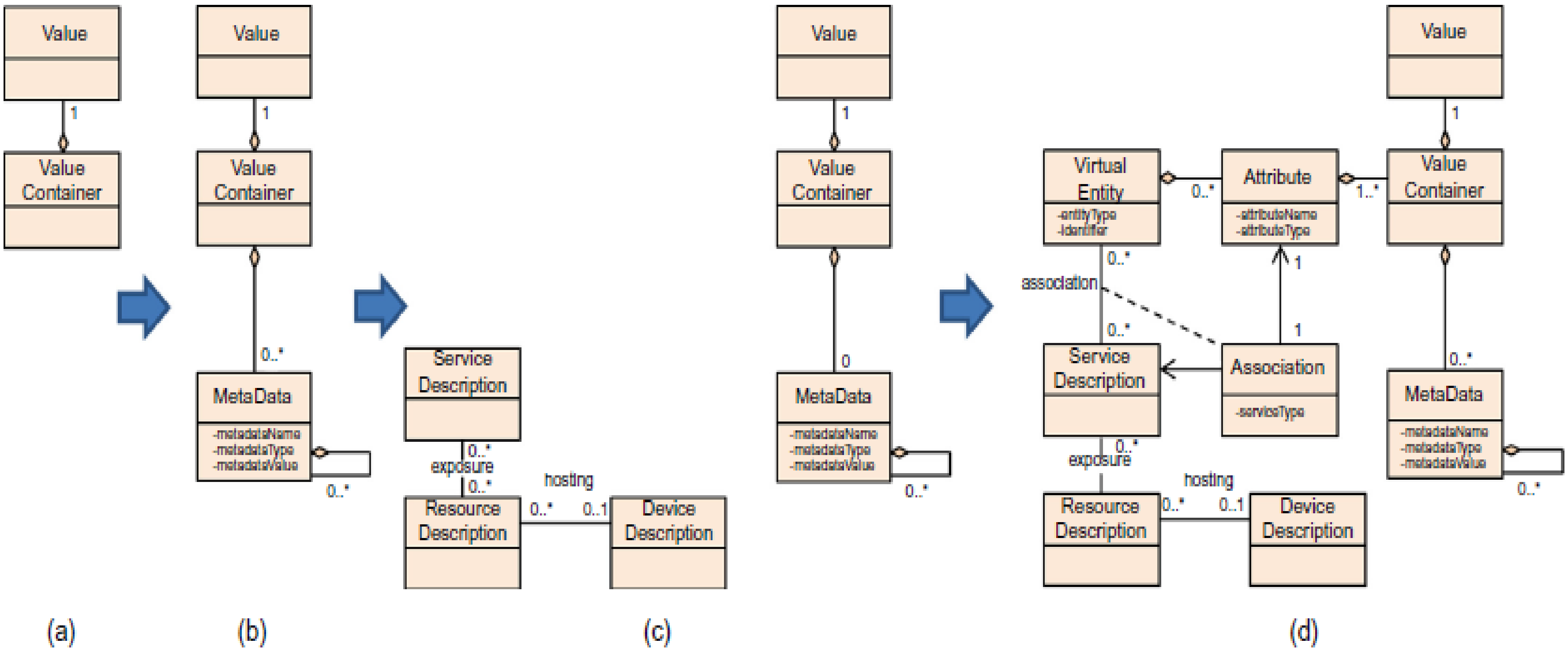


FIGURE 8.2

Information-enrichment process.

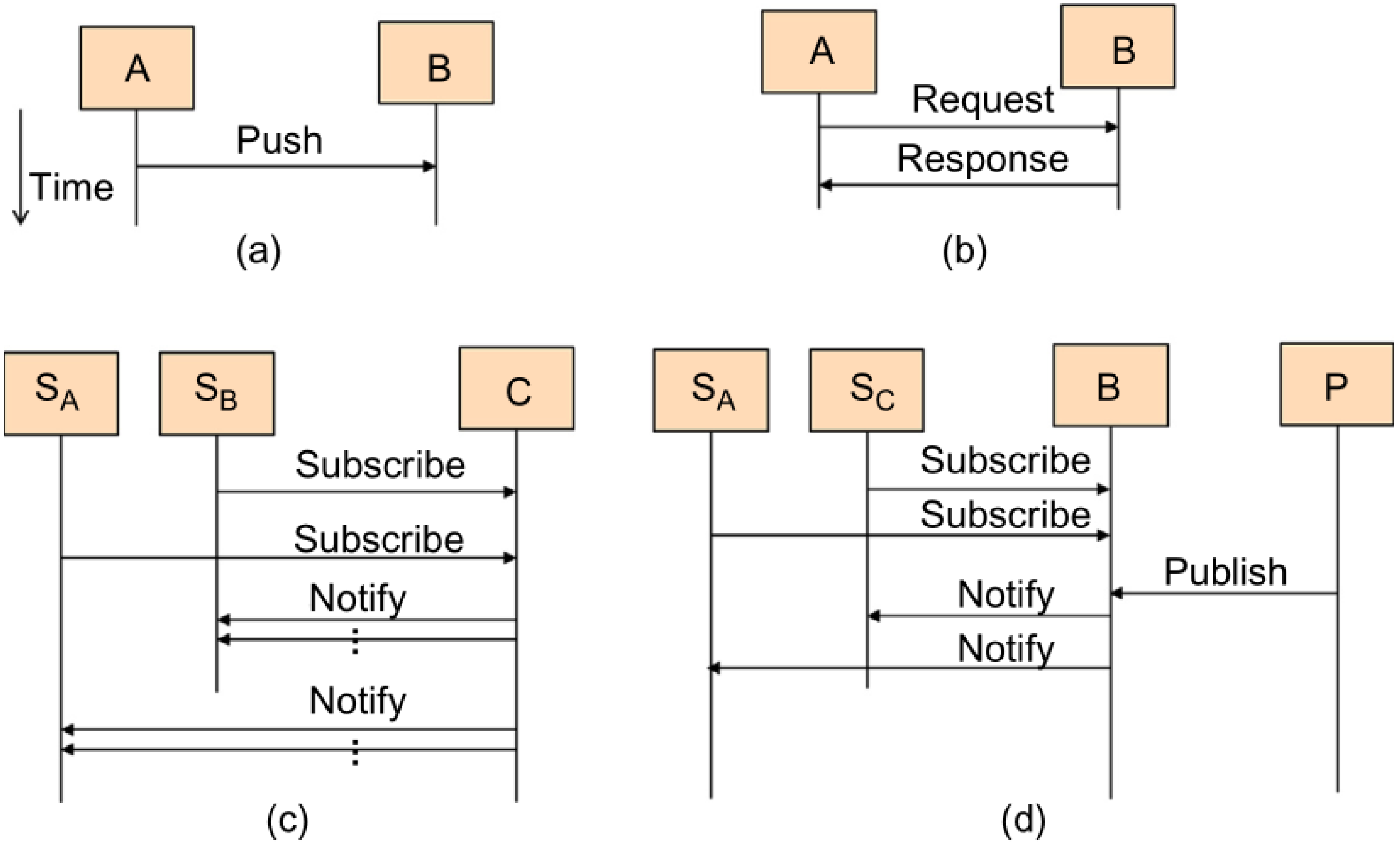
- At this point, the information is annotated with simple attributes such as location and time, and often this type of metadata is sufficient for certain IoT applications or for the use in certain larger systems.
- This enriched information becomes context information as soon as it is further associated with certain Physical Entities in the form of Virtual Entity attributes (simple or complex, static or dynamic).
- Further support information such as Associations between certain attributes and IoT Services further enriches the context information of the Virtual Entity (Figure 8.2d

- enrichment occurs in applications or larger systems that employ,
- for example, data analytics, machine learning, and knowledge management, which produces actionable information.
- Parts of the context and actionable information may be stored to an information store for future use.
- Actionable information flows into business processes that implement an action plan.
- Action plans push context information about Virtual Entities to associated IoT Services, to corresponding Actuation Resources, and finally to the real actuators that perform the changes in the physical world

Information handling

- An IoT system is typically deployed to monitor and control Physical Entities.
- Monitoring and controlling Physical Entities is in turn performed by mainly the Devices, Communication, IoT Services, and Virtual Entity FGs in the functional view.
- The presentation of information handling in an IoT system assumes that FCs exchange and process information.
- The exchange of information between FCs follows the interaction patterns as shown in next slide

Information Handling



(a)

(b)

(c)

(d)

Information exchange patterns.

Information Handling

- **Push:** An FC A pushes the information to another FC B provided that the contact information of the component B is already configured in component A, and component B listens for such information pushes.
- **Request/Response:** An FC A sends a request to another FC B and receives a response from B after A serves the request.
- Typically the interaction is synchronous in the sense that A must wait for a response from B before proceeding to other tasks, but in practice this limitation can be realized with parts of component A waiting, and other parts performing other tasks.
- Component B may need to handle concurrent requests and responses from multiple components, which imposes certain requirements on the capabilities for the device or the network that hosts the FC.

Information Handling

- **Subscribe/Notify:** Multiple subscriber components (S A , S B) can subscribe for information to a component C, and C will notify the relevant subscribers when the requested information is ready.
- This is typically an asynchronous information request after which each subscriber can perform other tasks.
- Nevertheless, a subscriber needs to have some listening components for receiving the asynchronous response.
- The target component C also needs to maintain state information about which subscribers requested which information and their contact information.
- The Subscribe/Notify pattern is applicable when typically one component is the host of the information needed by multiple other components.
- Then the subscribers need only establish a Subscribe/Notify relationship with one component.
- If multiple components can be information producers or information hosts, the Publish/Subscribe pattern is a more scalable solution from the point of view of the subscribers.

Information Handling

- **Publish/Subscribe:** In the Publish/Subscribe (also known as a Pub/Sub pattern), there is a third component called the broker B, which mediates subscription and publications between subscribers (information consumers) and publishers (or information producers).
- Subscribers such as S_A and S_B subscribe to the broker about the information they are interested in by describing the different properties of the information.
- Publishers publish information and metadata to the broker, and the broker pushes the published information to (notification) the subscribers whose interests match the published information.

Real-World Design Constraints

- The IoT allows for the development of novel applications in all imaginable scenarios.
- The technical design of any M2M or IoT solution requires a fundamental understanding of the specificity of the intended application and business proposition, in addition to heterogeneity of existing solutions.
- Developing an end-to-end instance of an M2M or IoT solution requires the careful selection, and in most cases, development of a number of complementary technologies.
- This can be both a difficult conceptual problem and integration challenge, and requires the involvement of the key stakeholder(s) on a number of conceptual and technological levels.

Devices and networks

- devices that form networks in the M2M Area Network domain must be selected, or designed, with certain functionality in mind.
- At a minimum, they must have an energy source (e.g. batteries, increasingly EH), computational capability (e.g. an MCU), appropriate communications interface (e.g. a Radio Frequency Integrated Circuit (RFIC) and front end RF circuitry), memory (program and data), and sensing (and/or actuation) capability.
- These must be integrated in such a way that the functional requirements of the desired application can be satisfied

Functional requirements

- Specific sensing and actuating capabilities are basic functional requirements.
- In every case with the exception of devices that might be deployed as a routing device in the case of range issues between sensing and/or actuating devices the device must be capable of sensing or perceiving something interesting from the environment.
- This is the basis of the application. Sensors, broadly speaking, are difficult to categorize effectively.
- Selecting a sensor that is capable of detecting a particular phenomenon of interest is essential. The sensor may directly measure the phenomenon of interest (e.g. temperature), or may be used to derive data or information about the phenomenon of interest, based on additional knowledge (e.g. a level of comfort).
- Sensors may sense a phenomenon that is local (i.e. a meter detecting total electricity consumption of a space) or distributed (e.g. the weather).

Sensing and communications field

- The sensing field is of importance when considering both the phenomenon to be sensed (i.e. Is it local or distributed?) and the distance between sensing points.
- The physical environment has an implication on the communications technologies selected and the reliability of the system in operation thereafter.
- Devices must be placed in close enough proximity to communicate.
- Where the distance is too great, routing devices may be necessary.
- Devices may become intermittently disconnected due to the time varying, stochastic nature of the wireless medium.
- Certain environments may be fundamentally more suited to wireless propagation than others

Programming and embedded intelligence

- Devices in the IoT are fundamentally heterogeneous.
- An application programmer must consider the hardware selected or designed, and its capabilities.
- The ability to reconfigure and reprogram devices is still an unresolved issue for the research community in sensor networks, M2M, and the IoT.

Power

- Power is essential for any embedded or IoT device. Depending on the application, power may be provided by the mains, batteries, or conversion from energy scavengers (often implemented as hybrid power sources).
- The power source has a significant implication on the design of the entire system.

Non-functional requirements

There are a number of non-functional requirements that need to be satisfied for every application. These are technical and non-technical:

Regulations

- For applications that require placing nodes in public places, planning permission often becomes an issue.
- Radio Frequency (RF) regulations limit the power with which transmitters can broadcast.
- Ease of use, installation, maintenance, accessibility

Physical constraints

- Can the additional electronics be easily integrated into the existing system?
- Are there physical size limitations on the device as a result of the deployment scenario?
- What kind and size of antenna can I use?
- What kind of power supply can I use given size restrictions

Non-functional requirements

Financial cost

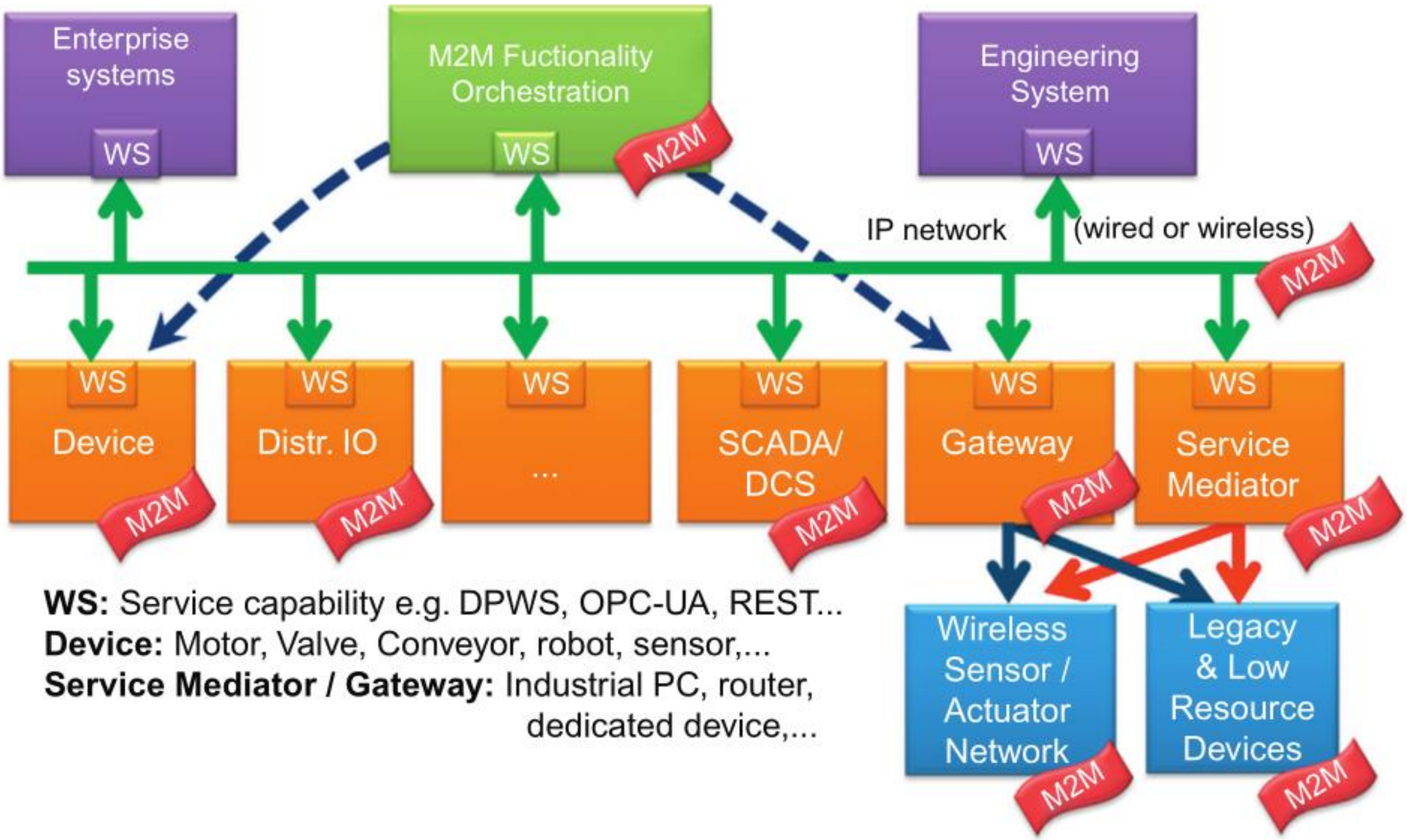
Financial cost considerations are as follows:

- **Component Selection:** Typically, the use of these devices in the M2M Area Network domain is seen to reduce the overall cost burden by using non-leased communications infrastructure.
- **Integrated Device Design:** Once the energy, sensors, actuators, computation, memory, power, connectivity, physical, and other functional and non-functional requirements are considered, it is likely that an integrated device must be produced.

M2M Service Oriented Architecture (SOA)-based integration

- The Service-Oriented Architecture (SOA) paradigm can act as a unifying technology that spans several layers, from sensors and actuators used for monitoring and control at shop-floor level, up to enterprise systems and their processes as envisioned in the diagram.
- This common "backbone" means that M2M is not limited to direct (e.g. proximity) device interaction, but includes a wide range of interactions in a cross-layer way with a variety of heterogeneous devices, as well as systems and their services.
- This yields multiple benefits for all stakeholders involved.

M2M Service Oriented Architecture (SOA)-based integration



WS: Service capability e.g. DPWS, OPC-UA, REST...
Device: Motor, Valve, Conveyor, robot, sensor,...
Service Mediator / Gateway: Industrial PC, router, dedicated device,...

M2M SOA-based integration.

M2M Service Oriented Architecture (SOA)-based integration

- Internet Protocol (IP)-based, and more specifically web technologies and protocols (e.g. OPC-UA, DPWS, REST, Web Services (WS), etc.), constitute a promising approach towards the fundamental goal of enabling easy integration of device-level services with enterprise systems overcoming the heterogeneity and specific implementation of hardware and software of the device.
- Surely industry specific requirements for security, resilience, and availability of near real-time event information needs to be effectively tackled.

M2M Service Oriented Architecture (SOA)-based integration

- The SOA-based vision is not expected to be realized overnight, but may take a considerable time depending on the lifecycle processes of the specific industry, and may be impacted by micro- and macro-economic aspects.
- Hence, it is important that migration capabilities are provided so that we can harvest some of the benefits today and provide a stepwise process towards achieving the vision.

Socrades: realizing the enterprise integrated Web of Things

- The SOCRADES project is a European research and advanced development project. Its primary objective is to develop a design, execution and management platform for next-generation industrial automation systems, exploiting the Service Oriented Architecture paradigm both at the device and at the application level.
- SOCRADES is a part of the Information Society Technologies (IST) initiative of the European Union's 6th Framework Programme.
- Socrades is driven by the key need for cross-layer M2M collaboration (i.e. at shop-floor level among various heterogeneous devices as well as among systems and services up to the Enterprise (ERP) level),
- SOCRADES proposed and realized SOA-based integration, including migration of existing infrastructure via gateways and service mediators
- The SOCRADES Integration Architecture (SIA), enables enterprise-level applications to interact with and consume data from a wide range of networked devices using a high-level, abstract interface that features Web Services standards.

Socrades

Various levels in Socrades are:

- **Application Interface:** This part enables the interaction with traditional enterprise systems and other applications.
- It acts as the glue for integrating the industrial devices, and their data and functionalities with enterprise repos and traditional information stores.
- **Service Management:** Functionalities offered by the devices are depicted as services here to ease the integration in traditional enterprise landscapes.
- Tools for their monitoring are provided.
- **Device Management:** Includes monitoring and inventory of devices, including service lifecycle management.

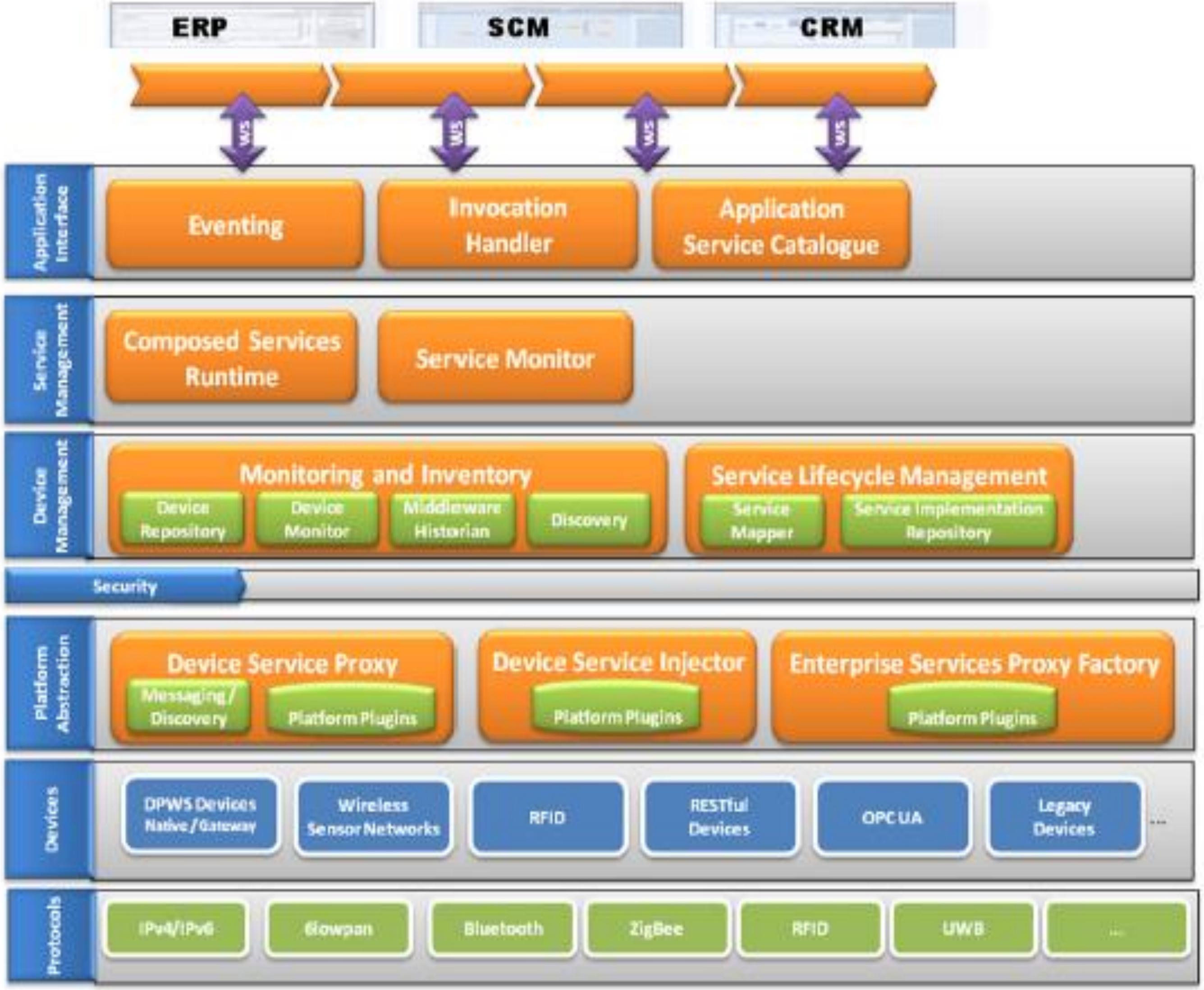


FIGURE 11.4

The SOCRADES Integration Architecture (SIA) enabling the coupling of (industrial) machines at shop-floor and enterprise systems.

Socrades

- **Platform Abstraction:** This layer enables the abstraction of all devices independent of whether they natively support WS or not, to be wrapped and represented as services on the higher systems.
- In addition to service-enabling the communication with devices, this layer also provides a unified view on remotely installing or updating the software that runs on devices.
- **Devices & Protocols:** These layers include the actual devices that connect over multiple protocols to the infrastructure.
- The respective plugins of course need to be in place so that they can be seamlessly integrated to SIA.