Domain framework for implementation of open IoT ecosystems

Milan Zdravković, Jelena Zdravković, Alexis Aubry, Néjib Moalla, Wided Guedri and João Sarraia

Faculty of Mechanical Engineering, University of Niš, Niš, Serbia; Department of Computer and Systems Sciences (DSV), Stockholm University, Stockholm, Sweden; CRAN, University of Lorraine, Nancy, France; CRAN, UMR7039, CNRS, University of Lorraine, Nancy, France; CRAN, UMR7039, CNRS, University of Lorraine, Nancy, France; DISP Laboratory, Université Lumière Lyon 2, Lyon, France; Luxembourg Institute of Science and Technology, LIST, Luxembourg; CTS, UNINOVA, DEE/FCT - Universidade NOVA de Lisboa, Caparica, Portugal

(Received 24 February 2017; accepted 20 September 2017)

The current Internet-of-things (IoT) hype, pushed by the unprecedented rate of the technological enablers’ innovation, is threatening to leave behind some major, not so obvious, unresolved issues. IoT platforms will extend existing enterprise information systems (EIS) infrastructures to encompass cross-domain sensing and actuating capabilities, thus introducing additional complexity and major risks to the implementation. Furthermore, IoT platforms are typically driven by models of the trivial complexity; they support very simple data structures and almost no business logic implementation. Finally, IoT systems are today managed centrally, which often means less openness, less flexibility and greater change management costs. In this article, we provide the overview of the scientific disciplines which could contribute to the resolution of the IoT implementation problem, namely requirements engineering, change management/continuous improvement, model-based systems engineering, system architecture design, interoperability and policy and regulatory aspects. Then, we identify the challenges of these contributions in the context of IoT and finally make an attempt to identify research directions which could have a significant impact. The discussion of the challenges and opportunities is illustrated by the proposed domain framework for implementation of open IoT ecosystems.

Keywords: Internet of things; interoperability; model-based systems engineering; maturity assessment; requirements engineering; multi-agent systems

Introduction

Internet of things (IoT) is considered as one of the 12 so-called disruptive technologies, by McKinsey Global Institute (Manyika et al. 2013), technological advances that will ‘transform life, business and the global economy’. The impact of IoT and its industry applications is multifaceted. New opportunities will be created to develop new services, to increase productivity, to improve decision-making, to solve critical societal problems and to develop new user experiences (Intel 2014).

While the technological innovation needed for developing individual IoT systems is already there, the challenges arising from implementing and maintaining its complex infrastructure, and creating interoperable IoT ecosystems, are still under research. These challenges start even at the adoption level, where lack of public policy and regulatory measures is an obstacle, both at macro-level (achieving full connectivity and interoperability) and micro-level (facilitating trustworthiness and incentivisation). The risks and uncertainty of IoT systems implementations increase when we consider the technical and organisational efforts and associated change, required by the enterprises. Even the adoption and implementation of conventional enterprise information systems (EIS) is still a big challenge for them. In 2010, the mean enterprise resource planning (ERP) systems’ implementation cost was $5.48 million, and the average implementation time frame was 14.3 months (Galy and Sauceda 2014). The failure rate of IT projects remains appealing. McKinsey and University of Oxford’s research have shown that ‘on average, large IT projects run 45% over budget and 7% over time, while delivering 56% less value than predicted’ (McKinsey report 2012). Due to increased complexity and interoperability requirements, it is expected that the failure rates of IoT projects’ implementations will be higher. According to IDC, 85% of existing devices worldwide are based on unconnected legacy systems (IDC 2013).

IoT platforms will extend existing EIS infrastructures to encompass cross-domain sensing and actuating capabilities, thus introducing additional complexity and major risks when considering the implementation. Also, even though there are already many cloud-based IoT platforms, great most of those are only big data aggregators, meaning that additional functionalities will need to be used by the other systems, resulting with probable interoperability risks. Furthermore, IoT
platforms are typically driven by models of the trivial complexity; they support very simple data structures and almost no business logic implementation. Finally, IoT systems are today usually managed centrally, which in the context of the heterogeneous environment of the IoT ecosystem often means more compromises on openness and greater change management costs. Thus, the problem of IoT implementation can be summarised to three questions: How to deliver and maintain the required (continuously changing) functionalities? How to deliver in an ‘open’ ecosystem of the heterogeneous components, technologies and standards? How to deliver in time and at cost?

**Methodology**

The answers to above questions are sought in the different domains, selected based on the following arguments. First, it is clear that enterprises need to have a wide understanding of the inner workings and impact of the IoT ecosystems, to be implemented. This understanding and even a shared agreement are established by using models. Second, models are developed as a result of the complex communication between different stakeholders. Such communication needs framework which ensures that modelled artefacts implement system requirements; this framework is typically established by using requirements engineering practices and tools. Third, the models specify some important technical decisions made after the requirements analysis. One of the most important ones is which architecture for IoT system to choose. IoT systems are inherently distributed; furthermore, multi-agent systems can decentralise IoT system and enable devices to make decisions locally. Fourth, interoperability is one of the greatest challenges in making a complex IoT ecosystem a reality. Though it may be considered also as a system requirement, interoperability is discussed separately because its impact spans multiple domains; it is a core feature of the IoT ecosystems. Both system requirements and architectural concepts must acknowledge that by taking into account interoperability issue in their formal definitions. Fifth, very complex change made by the implemented IoT systems poses the need to consider the evolution of one conventional enterprise to a sensing one. Such evolution must take into account maturity models and associated verification and validation processes. Special case of evaluation must take place in assessment of interoperability, as the most critical requirement for the open IoT ecosystems. Finally, this openness implies a strong need to take into account different societal, policy and legal aspects in their implementation. The above domains are interrelated, and they form the proposed domain framework for addressing IoT implementation problem (see Figure 1).

The relevance of the proposed domain framework for IoT implementation problem is considered as a priori hypothesis of the research behind this paper. However, the research is exploratory, as it seeks to identify finer grained concepts behind each of the domains and identify relationships between them, thereby producing more practical framework. This framework aims to provide a blueprint or even checklist (not methodology) for IoT ecosystem implementation by considering all its relevant factors and relationships between those. The effort is made collaboratively, by synthesising contributions of the experts (co-authors) in each of the proposed domains, based on the literature review. First, the relevant concepts in each of the fields are explained, and more mature research, close to or already on market, is referenced. It is

![Figure 1. A domain framework for addressing IoT implementation problem.](image-url)
followed by the discussion of the challenges and prospects in the frontiers of the selected fields’ research for addressing
the implementation problem.

Societal and policy aspects are discussed separately in the context of the potential legal and societal implementation
constraints; they have impact on each of the remaining domain framework elements; they are characterised by the socio-
logical and governance factors which are emerging as innovative response of the society to pervasive IoT, unmatched to
the previous practices.

**Background research**

In this section, each of the domain framework elements is discussed with objective to introduce key concepts and their
relationships. Also, where applicable, main technologies, already on or close to market, are highlighted, and short over-
view of the research relevant for IoT with highest innovation potential is given.

**Model-based system engineering**

International Council on Systems Engineering (INCOSE) defines model-based system engineering (MBSE 2007) as ‘the
formalised application of modelling to support system requirements, design, analysis, verification, and validation activi-
ties beginning in the conceptual design phase and continuing throughout development and later lifecycle phases’. MBSE
engineering process is often integrated with software engineering. Model-driven development (MDD) is the process in
which problem-level software abstractions are systematically transformed to their specific implementations. One of
the better known MDD initiatives is model-driven architecture (MDA) of object management group (OMG). Developed and
maintained jointly by INCOSE and OMG as an extension of the unified modelling language (UML) standard, system
modelling language (SysML) is today the most used language for the systems specification, analysis, design, verification
and validation.

When model-based implementation of IoT systems is considered, the literature review reveals three characteristic
approaches in the different stages of systems engineering. At the design level, widely accepted MBSE standards –
UML/SysML languages and tools – are being used. At the level of programming a controller (IoT platform), researchers
and practitioners use more implicit formalisms, namely the domain-specific languages (DSL), to enable even domain
experts to develop IoT applications. Finally, to facilitate interaction modelling in a heterogeneous environment and thus
to resolve interoperability problem, many researches rely on the formal models, namely ontologies (see Figure 2).

In a joint initiative of OMG and IIC, SysML is a key part of IoT MBSE standardisation stack. IoT system design
tools, based on SysML, have also started to emerge.

Domain-specific languages (DSL) are used to model application logic in controllers of the IoT systems, hidden due
to a widespread use of specific, often proprietary software development kits (SDK). Patel and Cassou (2015) proposed a
framework to a model-based application development for IoT. Framework also includes a conceptual model, which
defines domain-specific, functionality-specific, deployment-specific and platform-specific concepts. This conceptual
model is a starting point for defining a DSL for IoT. Garcia et al. (2014) have developed end-to-end solution for IoT
platform which includes Midgar IoT platform, DSL for abstracting the application generation problem and graphical tool
supporting the use of the proposed DSL. Harrand et al. (2016) have proposed ThingML, a modelling language and tool
which supports forward engineering of the distributed, heterogeneous systems.

One IoT system cannot be considered in isolation from the others, with which it interacts. Also, one specific device
may have roles in two or more different IoT systems, hence a need to consider explicit modelling interactions and inter-
operations of many IoT systems. To address this challenge, devices ‘will need to be consistently and formally repre-
sented and managed, registered, aligned, composed and queried through suitable abstraction technologies’ (Kotis and
Katasonov 2013). For this, we need an ontology and exposure of devices’ interfaces by using REST or SPARQL

![Figure 2. Technology stack for MBSE of IoT systems.](image-url)
endpoints. Ontology aims at formally describing IoT entities, for the purpose of their discovery, querying and clustering into sub-systems.

Finally, different viewpoints to the IoT entities will be needed. For example, while most of the current research focuses on explicit semantic modelling of IoT resources, Wang et al. (2012) proposed a formal ontology which highlights accessibility and utilisation of those resources, from the SOA point of view. This ontology can be used for IoT service discovery, testing and dynamic composition.

Requirements engineering
MBSE is firmly related to requirements engineering (RE), because the latter delivers implicit information which is then, in modelling phase transformed to explicit constructs, understood by the computers.

In the business analysis community, a requirement describes a condition of the current or a future state of any aspect of an enterprise (Zdravkovic et al. 2014). A basic premise for a requirement is to be agreed by all interested in its fulfilment, so-called stakeholders. As EIS pervade every aspect of today’s organisations, a majority of initial business requirements become the goals for those systems. Once they are analysed and decomposed for a possible support by systems, they are typically transformed to models and the architecture for systems, determining thus both their functionality and their quality (aka non-functional) aspects.

The main objective of requirements engineering for systems is to manage requirements entirely – take into consideration all requirements from the stakeholders – and correctly – a system should reflect the way of working of the business which it is supporting. It is therefore widely accepted to follow a process for dealing with requirements systematically, which includes the activities for their elicitation, documentation, analysis/negotiation, validation and change management (Kotonya and Sommerville 2002).

Because the demand for increasing flexibility and productivity of organisations is constantly requiring shortening of EIS development life cycles, the RE process has over a time evolved from a traditional sequential execution of its activities to more agile. Consequently, a number of methods for iterative and incremental system development have emerged (Leffingwell 2011). Such methods propose practices for shortening system’s development, frequent releases, simple design and minimal documentation. As for the RE process, that in particular means: interactive and group-based elicitation of requirements, reduced documentation in the form of user stories or meeting minutes, development of test cases, quick responsiveness to change, etc.

Following the above, in the problem domain of IoT, as for many others, it is important to leverage the orchestration of the RE activities according to the size and criticality of the project. Even though requirements engineering is a universal discipline, some recent research studies (Zambonelli 2016) have emphasised that for IoT, the key abstractions and concepts are (a) goals – desirable situations or state of the affairs that should be activated and which should be decomposed to system requirements – and (b) identification of stakeholders and users who will manage or use systems’ functionalities and from which functional requirements should be elicited. In addition, a number of studies advocate importance of elaboration of non-functional requirements, such as security.

System architecture
As it was previously highlighted, the decisions, related to functional and non-functional requirements, are made explicit in the modelling phase. Then, these explicit constructs need to be forward engineered in the selected run-time environment. The choice of its architecture is largely dependent on the input from the modelling phase, but it also will consider innovation in EIS architecture domain.

The IoT system is inherently distributed. It uses different communication protocols to coordinate different sub-systems which acquire (sensors), pre-process (IoT gateways), store (clouds), process, visualise, interpret (IoT platforms) data and then act based on those interpretations (actuators). The agent-based EIS implements the concepts of distributed, decentralised systems that can deal with flexibility, integrate dynamic conditions and be open to system components which come and go. Agent-oriented methodologies started to appear in the nineties (Wooldridge, Jennings, and Kinny 2000). The first challenges targeted handling of continuously changing requirements (Brazier et al. 1995). The recent works proposed architectural patterns, middleware for distributed agents, evaluation model for distributed multi-agent systems, etc. (Weyns 2010).

Agent-based node in IoT systems is independent, self-governing software- and hardware-integrated system. It is capable to sense, and to interpret the sensation, make the best informed decision on that interpretation and finally act upon it (see Figure 3).

In order to organise the interaction between agents, two composition approaches are possible:
Centralised agent-based approaches: the composition of sensors is pre-defined. Each agent is aware of the decision capabilities of its neighbours. The reliability of the agent network depends on the reliability of each one of its components.

In decentralised approaches, agent-based node in IoT systems is independent, self-governing software- and hardware-integrated system. These approaches are faced with some potential performance issues, such as convergence of the ecosystem, robustness, time needed to achieve the balance and scalability.

In the future IoT, smart objects will be the fundamental building blocks for the creation of cyber-physical smart pervasive systems. The implementation of smart object-oriented IoT is complex challenge as distributed, autonomous and heterogeneous IoT components at different levels of abstractions and granularity need to cooperate among themselves, not only with conventional networked IT infrastructures, but also with humans.

**Maturity models**

In open environments, such as IoT ecosystem, it is crucial not only to continuously assess the capability of a system (and its stakeholders) to maintain its operation at the agreed quality levels, but also to evolve, sometimes even in very short time frames, on demand.

Maturity models are an established means to systematically document and guide the development of organisations using archetypal capability levels (Lahrmann et al. 2011). The term maturity model was popularised by the Software Engineering Institute (SEI) when they developed the capability maturity model (Paulk 1993). In information systems and management science fields, maturity models (MM) have been applied both as an informed approach for continuous improvement (Ahern, Clouse, and Turner 2003) and as a means for self-assessment or third-party assessment (Lahrmann et al. 2011).

Mettler (2011) found more than 100 maturity models in different domains in the literature. They can be captured qualitatively or quantitatively in a discrete or continuous manner (Kohlegger, Maier, and Thalmann 2009). They typically include a sequence of levels (or stages) that form an anticipated, desired or logical path from an initial state to maturity (Becker, Knackstedt and Pöppelbuß 2009).
A wide range of maturity assessment models have been developed by both practitioners and academics over the past years to ascertain and measure different aspects of social and technical systems 'maturity'. For instance, a business process maturity model (BPM) may assess how capable an organisation is in modelling its processes or in running its processes without errors; a maturity model for enterprise interoperability (MMEI) may assess how ready an enterprise is able to interoperate with another one (Guédria, Naudet, and Chen 2015).

Findings from an assessment are typically transformed into a roadmap for improvement. The roadmap is realised by actions which are expected to result in better performing systems. This chain of activities is performed continuously as an application of Deming’s ‘Plan-Do-Check-Act’ cycle (Madu 1998) for excellence of organisational/system performance (Tarhan, Turetken, and Van Den Biggelaar 2015).

To overcome growing uncertainty in industries, stemming from challenges for a new kind of intelligent, networked and agile value chain driven by IoT opportunities, new maturity models have been developed to provide guidance and support to align business strategies and operations. For instance, Schumacher, Erol, and Sihn (2016) developed a maturity model and tool to systematically assess manufacturing companies’ state-of-development in relation to the Industry 4.0 vision. The TDWI research has also developed a maturity model to enable organisations to gauge their readiness for IoT and to compare themselves against others with IoT initiatives (Halper 2016). Lichtblau et al. (2015) proposed IMPULS – Industries 4.0 readiness maturity model. The connected enterprise maturity model (Rockwell Automation 2014) is a technology-focused assessment model in four dimensions, a part of a five-stage approach to realise Industry 4.0. Integrated IoT capability maturity model (Vachteryté 2016) is used to assess the company’s progress in IoT implementation in five levels and three dimensions.

**Interoperability**

In an open, heterogeneous world of IoT, interoperability of devices is considered as one of the most difficult issues to resolve during implementation.

ISO/IEC 2382 vocabulary for information technology defines interoperability as ‘the capability to communicate, execute programmes or transfer data among various functional units in a manner that requires the user to have little or no knowledge of the unique characteristics of those units’. In a more broad sense, Vernadat (1996) defined interoperability as ‘the ability to communicate with peer systems and access the functionality of the peer systems’. Interoperability lies in the ‘Integration Continuum’ (Molina et al. 2007) between compatibility and full integration. While integration assumes functional interdependence, interoperability of systems means that they work in their domains while invoking each others’ functionality.

In IoT, system interoperability is meant to be facilitated by application layer protocol devices used to communicate over both persistent and intermittent network connection. Some of the most often referred application layer protocols are shortly presented below:

- Representational state transfer (REST) is architectural style, rather than protocol, which implements synchronous request/response HTTP functions to facilitate exchange of XML and JSON messages. Although it is widely used, it is unlikely that it will become a dominant protocol due to its inconvenience for resource-constrained devices.
- Constrained application protocol (CoAP). Although it conforms to request/response REST style, it is based on UDP and therefore lightweight. It is realised on two sub-layers: interaction sub-layer implements a subset of HTTP functions (GET, POST, PUT, DELETE, etc.), while messaging sub-layer facilitates asynchronous, reliable interactions over UDP, by implementing confirmable, non-confirmable, reset and acknowledgement types of messages.
- Message queue telemetry transport (MQTT). MQTT implements publish/subscribe pattern to address mainly reliability and low bandwidth issues (even though it runs on TCP). In MQTT interaction, clients are publishing information to a broker (server) on the specific topic, while subscribers receive a message, every time a new update to a specific topic to which they are subscribed is published.
- Extensible messaging and presence protocol (XMPP). Unlike MQTT which is relatively new protocol, XMPP is already well established, initially implemented for chat and message exchanges. It implements both synchronous request/response and asynchronous publish/subscribe patterns. However, it does not support QoS options, and it uses XML messages and therefore creates additional overhead in bandwidth (XML tags) and processing (XML parsing).

At the semantic layer, the problem of interoperability was so far dealt by the IoT platforms or middlewares, in centralised manner. The dominant approach relies on the assumption that devices interact by using SOA approach. In such architecture, designers (but also devices) can dynamically discover, select and use services running on devices in their reach (Guinard et al. 2010). These services, namely devices’ capabilities, can be semantically represented, so the process
of discovery and selection is done by using SPARQL queries (Song, Cardenas, and Masuoka 2010). Recently, the works that recognise that scalability, flexibility and other issues cannot be resolved with centralised approach have started to emerge. Katasonov et al. (2008) proposed that each of the devices must be represented by (and connected to) its virtual counterpart, which is implemented as autonomous software agent. IoT platform is then only a run-time environment for these software agents.

**Challenges and opportunities**

As it was shown above, the scientific community has already acknowledged high relevance of the framework elements for IoT systems implementation. In this section, we discuss on the relevant challenges of the respective topics and opportunities for increased impact to resolving implementation problem. Both are then structurally highlighted, in Table 1.

**Model-based systems engineering**

Paredis (2011) has identified several challenges for MBSE and classified them into those related to efficiency and rationality. The biggest efficiency challenge is the cost (of time consumed and errors made) of manual creation of models, including maintaining dependencies between different model views. Rationality challenges are related to choices made in very heterogeneous environment of non-synced models, languages, beliefs and preferences of the system engineers.

When MBSE of IoT systems is considered, several challenges are highlighted. One of the most notable is the model mapping and transformation, arisen from the complex interoperability requirements in IoT ecosystem. Furthermore, even if different model views and components are well integrated, it is not possible to validate (and thus, to maintain) the integrated model consistency with the current range of modelling tools. Finally, lack of formalisms for representing business logic models for IoT platforms remains an obstacle for implementation cost, since the application logic of the IoT applications is hard-coded and thus resistant to changes.

Use of formal models or semantically annotated models in MBSE and MDD is proposed as a potential solution to each of the challenges above. Models mapping is so far considered as exclusively human task, time-consuming and error-prone. With an increasing rate of existing ontologies consolidation, as well as advances in semantic matching tools, this activity could be automated to a certain extent. The model validation problem could be resolved by formal specification techniques (FST). FST aims at restricting the modelling viewpoints, with objective to provide analysis, transformation and generation tools. A common approach is to translate a modelling view (UML class model) to a form that can be analysed using a particular formal technique (McUmber and Cheng 2001). This analysis can involve checking of the consistency, completeness and dependability. Thus, reasoning on the formal specification of one system can be further used to prove that all actions will result in discrete set of states; that all system properties are bound and that error states are unreachable.

Use of formal models and associated methods (semantic annotation, ontology matching, etc., see Figure 4) is one step towards ontology-driven IoT systems, which uses formal models in a run-time. Currently, run-time models are considered as assets which are used to monitor and verify particular aspects of the run-time behaviour of the IS (Bencomo, Blair, and France 2007). Run-time models are then used by the agents responsible for managing the run-time environment and for adapting and evolving the software during run-time. Ontology-driven systems are future systems that take a step further forward by providing the theoretical and technical background for run-time interpretation of a framework of the formal models, where this framework formally describe IoT application data, inner structure (composition of devices) and business logic, in the context of the given domain (represented by the domain ontologies).

**Requirements engineering**

From the requirements engineering perspective, the overall challenge concerning integration of IoT with EIS is the difficulty of specifying requirements for diverse IoT components that eventually should be used by various users and systems in various contexts. Given that requirements specification is the key task in which stakeholders and requirements are to be identified, specifically for IoT, the challenges include: identification of relevant sources of requirements and requirements themselves and identification of ‘innovation’ requirements.

If all relevant sources of requirements (stakeholders, objects and systems) are not identified, they will not be considered in the elicitation and this will eventually lead to an incomplete requirements specification. A challenge thus is first
| Requirements engineering | Model-based system engineering | Multi-agent systems | Interoperability | Maturity models | Policy and regulation aspects |
|---------------------------|---------------------------------|--------------------|------------------|----------------|------------------------------|
| **Challenges**            |                                 |                    |                  |                |                               |
| - Scoping (domain, context and sources) | - Model mapping and transformation | - Model validation | - Data management | - Heterogeneity | - Lack of formal theoretical basis |
| - Requirements identify/acquisition | - 'Innovation’ requirements | - Business logic modelling | - New domains application | - Agreements to interoperate difficult to establish | - Sharing reference model |
| - Connectivity | - Model mapping and transformation | - Cloud infrastructures | - New service development | - Limited expressiveness of DL-based languages | - Domain dependability |
| - Privacy | - Automatic semantic matching | - Trusted, smart data | - Abstract, formal models of the devices capabilities (SSN Ontology-based) | - Intelligent agents with capability to interoperate | - |
| - Security | - Formal specification techniques | - Unified access control schemas | - Intelligent agents with OWL extensions | - Data ownership policies | - |
| - Standardisation | - Transforming *ML to FST | - On demand, dynamic deployment of IoT systems | - Formal models | - Opening ‘white space’ of the spectrum | - |
| - Data ownership | - Run-time models | - Self-management capability | - Combining explicit domain formal models with implicit application ontologies | - New products for M2M connectivity | - |
| **Opportunities**         |                                 |                    |                  |                |                               |
| - Conceptualisation of new classes of requirements | - Semantic annotation of models | - Cloud infrastructures | - Abstract, formal models of the devices capabilities (SSN Ontology-based) | - Intelligent agents with OWL extensions | - |
| - New methods for RE, goal prioritisation and change management | - Automatic semantic matching | - Trusted, smart data | - Intelligent agents with OWL extensions | - Data ownership policies | - |
| - Acquisition of requirements from (big) data | - Formal specification techniques | - Unified access control schemas | - Intelligent agents with OWL extensions | - Data ownership policies | - |
| - Elevating people empowerment | - Transforming *ML to FST | - On demand, dynamic deployment of IoT systems | - Formal models | - Opening ‘white space’ of the spectrum | - |
| - | - Run-time models | - Self-management capability | - Combining explicit domain formal models with implicit application ontologies | - New products for M2M connectivity | - |
| | | | | - Traffic prioritisation | - |
| | | | | - Privacy-by-design, Security-by-design | - |
| | | | | - Open standards | - |
| | | | | - Data ownership policies | - |

| Technologies | |
|--------------|---------------------------|
| - RE Framework and tools | - Semantic matching algorithms and tools |
| - MAS development/deployment tools | - Application protocols |
| - OWL reasoners with extended capabilities | - Model-driven run-time assessment tools |
| | - IPv6 |
| | | - Network layer protocols |
to identify all stakeholder groups, which, for example a shop floor, include operators who use IoT, maintenance engineers, production supervisors, programmers, as well as front-office managers who are getting information derived from the IoT data. The different physical objects and systems could be sources of requirements in terms of their mutual interaction.

Identification of requirements assumes their elicitation from the identified stakeholders where the key issue is to obtain a complete specification. The different techniques – from interviews, to observational studies and prototype development – could be implemented to exhaust all scenarios for all possibly different conditions of use, as well as other needed data – about uptime, internal alarms, operational status signals, energy usage and many other performance characteristics and parameters. Even a higher challenge may concern elicitation of non-functional requirements, such as security and privacy. Furthermore, continuous device availability and processing performance are important. As streams of data going from IoT may be massive, capacity of network and (cloud) storage are important quality requirements that need to be tested and monitored for fulfilment.

Identification of ‘new’ requirements as an ‘innovation force’ is a continuous challenge for IoT stakeholders who should be organised to brainstorm about new ways of use of existing IoT, as well as on how to develop new requirements to improve the alignment between emerging company strategies with new technological solutions.

To overcome the outlined challenges, it is important to consider frameworks which, referring to the main RE activities, should be capable for managing the requirements by taking into account the following:

- Reliable methods for identifying all relevant sources of requirements
- Reliable methods for identifying all different contexts in which IoT will operate – environmental, security, regulatory and other.
- Methods for setting up and prioritising goals for non-functional (quality) requirements.
- Methods and rules for change management due to new incoming operational conditions, changing company policies, technical innovations and others.
- Consideration of new classes of requirements to support better decision-making through new types of gathered information as a key source for creating a connected smart factory in which machines and people become more efficient (fewer mistakes, less waste, etc.) by newly established strategic objectives and activities of decision-makers.
- Achieving higher levels of automation of IoT use by eliciting requirements to improve sensing, analysis, prediction and control.
- Use of IoTs’ data as a source for eliciting new requirements for manipulation/analysis.
- Creating patterns for structuring of IoT data to find reusable patterns of behaviour (‘make sense of data’).
- To further elevate people empowerment by specification of new requirements for the purpose of smarter and easier use, education and management about IoT.

Figure 4. Formal framework for multifaceted view to IoT ecosystem.
**Multi-agent systems**

Explicit coupling between agent-based concepts with IoT is quite recent; the first research papers started to appear in 2013. Based on the literature analysis, we can identify three main orientations with significant research production. Those orientations also highlight the specific challenges:

- Data management: managing IoT data in order to maximise its exploitation in decision-making
- Domain application: the implementation of agent-based IoT in different application domains and
- New service development: the exploitation of IoT data in order to create new infrastructural services.

With MAS in IoT ecosystems, especially those whose agents exhibit data interpretation and autonomous decision-making, the quantity of data which needs to be stored becomes even larger, while management (now, decentralised) of that data becomes complex, thus requiring specific architecture (Manate, Munteanu and Fortis 2013). In addressing the challenge of massive data and information overload, Sriram and Sheth (2015) introduce the concept of ‘smart data’ ‘which is increasingly making sense in conveying how all the volume, variety, velocity and veracity challenges of physical, cyber and social big data need to be managed to derive its value’. In open IoT ecosystems, localised interpretation also poses the need for the implementation of a trust model, which will ensure that proper decisions are made in environment of uncertain credibility, high reliability and security concerns. The trust model can be complemented with appropriate unified access control schema (Rivera et al. 2015).

MAS facilitate the implementation of IoT systems in heterogeneous and dynamic environments. This has been already validated in the different domains. In planning and control (Herding and Mönch 2016), the MRP process is upgraded with using agent for production operations and lot planning to provide the decision-making process with several alternatives to be dynamically selected. This mechanism would help avoiding machine breakdowns, inappropriate lot sizing, etc. Schwartz et al. (2016) add virtual agents to the collaboration scenarios. They upgrade collaboration artefact with agents and use event-based middleware to select the best agent composition satisfying the collaboration requirements.

Finally, MAS approach to developing IoT ecosystems could open doors for the service innovation. MAS can help to deploy IoT ecosystems by using smart phones on demand, dynamically (e.g. by tracking availability and usability of services they provide) and to facilitate their adaptive behaviour (Verma et al. 2014). Another possible innovation opportunity lays in the convenience of MAS approach for ensuring self-management capability of IoT systems, by realising the context-awareness and self-adaptation properties of the individual agents (Ayala et al. 2015).

**Maturity models**

Maturity models describe essential attributes that are expected to characterise the assessment at a particular maturity level. By comparing a system’s characteristics and attributes with the target maturity level, the strengths and weaknesses are identified in order to characterise the as-is situation and plan improvement actions: first, establishing goals for the improvement and then, using best practices to achieve them.

The application of maturity model approach will make it easier to establish goals for improvement and identify opportunities. It will mainly provide the following benefits:

- A starting point: It is very important for any system to identify its as-is situation (current state) in order to set up actions that are necessary to achieve the defined objectives.
- An improvement path: Having a framework of best practices, based on prior experience of knowledgeable people, is very useful to build the improvement path from the as-is situation to the to-be one, with details of the needed steps to improve a given situation.
- A reference model: using the same maturity model implies sharing a common glossary and ensures that people are using the same language and a shared vision.

Despite the different benefits of maturity model approaches, their definition and their levels are mainly based on the experience of ‘knowledgeable people’ of the domain and they lack a formal theoretical basis. They usually contain only very little information on the system/process dynamics and consider a static standard evolution instead of a context-aware situation where the context and the properties of the system are taken into account. Moreover, most of the maturity models propose standard best practices to reach higher maturity levels and improve current situations. This can be challenging, as there are no best practices regardless of the context. Given that best practices cannot be applicable in all contexts (or that the system has no will to apply them) and that a success history cannot be considered as a pattern for other ones, we cannot talk about practices that everyone should follow.
Starting point and a reference model is considered already as part of the overall model framework, which is universal sensing enterprise asset, addressing all issues, including maturity assessment. Domain and/or context dependability of the maturity model could be addressed by separating the explicit and generic continuous improvement and maturity assessment concepts from the implicit ones, coming from the specific domain. Thus, we can foresee the development of maturity assessment reference ontology, which generically and formally defines improvement paths and maturity levels. Such ontology can be used to formally assess above, in specific contexts only when combined with specific domain ontology and an application ontology, which makes possible to apply the generic assessment and continuous improvement concepts to the specific domain. Then, maturity model is considered as an instantiation of the above-mentioned application ontology.

Interoperability

Interoperability is sine qua non for the sustainable development of open IoT ecosystems. With the development of multitude of protocols for device communication, many researchers assume that interoperability problem can be reduced to mapping and transformation challenge. Also, current perception of interoperability often assumes that, before interoperation takes place, there has to be an agreement between interoperating parties on how they will interoperate.

The above assumption cannot hold in open IoT ecosystems with uncertain heterogeneity, neither interoperability can be reduced to simple translation. Actually, interoperability is often related to the federated approach (ISO/IEC 2382), which implies that a few or, in ideal situation, no pre-determined assets for interoperations are assumed. In reality, this means that in order to interoperate with another device, each device must sense, perceive, interpret and understand data, sent from another device and act (operate) upon this understanding. Those capabilities are attributes of semantic interoperability, and they are the building blocks for intelligent behaviour. In fact, Zdravković et al. (2017) defined semantic interoperability capability as ‘complex ability to sense and perceive a stimulus, namely a message by another system in its environment (based on the perceptual sets that include interoperating entities’ experience, domain knowledge, motivational, emotional and environmental factors), to make an informed decision about this perception and consequently, based on this decision, to articulate a meaningful and useful action or response’.

First step towards making such capability a reality is to abstract the heterogeneity of devices, so one device can better understand the capabilities of another. One of the most prominent works in this area was related to developing W3C Semantic Sensor Network (SSN) Ontology, a formal OWL DL ontology (Compton et al. 2012) for modelling sensor devices (and their capabilities), systems and processes. It unfolds around the central pattern that relates what the sensor observes to what it detects. While the latter is determined on the basis of its capability, namely accuracy, latency, frequency, resolution, etc., and a stimulus, the former is related to the concepts of features of interest, their properties, observation result and sampling time, etc.

In computing, the problem of understanding can be reduced to inference of the new explicit knowledge, based on the perception of sensation, domain knowledge and formal expression of agent goals. Thus, we can foresee the interoperating engine, which is basically a formal reasoner with extensions. All the above-mentioned formal descriptions are by default expressed by using Semantic Web stack of languages (RDF/RDFS/OWL), based on description logic. However, their expressiveness is quite limited, especially when considering representation of vagueness and uncertainty, and reasoning context. There are already some works towards resolution of the above issues, though their applicability in realistic conditions (reasoning over big data) is not yet tested. Based on Bayesian networks, Costa and Laskey (2006) formally defined a probabilistic ontology and developed the OWL extension (PR-OWL). In probabilistic ontology, each axiom is annotated with a probability that can now be computed for each of the executed queries (Bellodi et al. 2011) affecting this axiom. The contextual approach to reasoning argues about its opportunistic nature. McCarthy and Buvac (1997) established the basic relation $\text{ist}(c, p)$, meaning that the proposition $p$ is true in the context $c$, and $\text{value}(c, e)$ designating the value of the term $e$ in the context.

Policy and regulatory aspects

While IoT ecosystems will directly benefit from already established technologies and principles in the above domains, required level of innovation related to the policy and regulatory aspects, as enablers of IoT, is much higher.

A vision for the nationwide IoT ecosystems demands appropriate policy principles which will address the societal challenges of all pervasive M2M connectivity. These principles form the regulatory framework, and they could be classified into following categories: (a) connectivity; (b) privacy; (c) security; (d) standardisation and (e) data ownership (see Figure 5).
The main concern of the policy is to ensure that connectivity is ubiquitous, affordable and high speed, over-licensed and unlicensed spectrum. Addressing issue should be faced by IPv6, whose adoption should be considered as IoT enabler of the highest national priority and its roll-out at the national levels should be encouraged by the governments (ISOC 2016). Recently, UK developed a spectrum strategy which is aiming to exploit so-called ‘white space’ of the spectrum – underused portions of radio frequency (RF) spectrum for wide commercial use, namely for new kinds of mobile technologies, more bandwidth and new services. One of the short-term solutions for all pervasive connectivity is using cellular frequencies. Mobile operators should be encouraged to develop new products for M2M connectivity, special SIMs and accounts suitable for large M2M users (Brown 2015). Ideally, switching (between operators) and roaming services should not be provided at extra cost. A reasonable and effective inter-carrier cost structure is important prerequisite for continuing growth of IoT ecosystem (Halliday and Lam 2016). Finally, effective implementation of IoT ecosystem needs to consider the traffic prioritisation in cases where the reliability is core feature of the service (e.g. health monitoring devices).

Privacy and security are the centrepoints of the IoT policy considerations. The basic rights, such as consumer consent and notice, right of deletion and right to be forgotten will remain important. Still, other privacy principles will emerge, for example, related to accountability of service providers for use of data across clouds or networks. Data transmission from one to another jurisdiction will occur more frequently in the connected IoT ecosystem, often based on mashups and cloud services. If the data protection laws in these jurisdictions are different and especially if those laws do not consider the possibility of transmission, cross-border IoT ecosystems adoption rate would be affected (ISOC 2016).

Mainly due to lower computational capacity, securing the IoT applications is quite a different challenge than the one related to conventional software security. Devices need to have more processing capabilities, to be designed for much longer execution, and security updates must be much easier to install. Recently published HP’s study (HP 2014) revealed that 70% of the most commonly used IoT devices contain vulnerabilities (in average, 25 per product). The lack of manufacturers’ motivation to consider device security as an important issue causes a high pressure to regulators. However, Federal Trade Commission (FTC) believes that self-regulation is better than regulation in case of IoT systems.

| Connectivity | Ubiquitous | (unlicensed spectrum) | Cellular frequencies | Switching and roaming |
|--------------|------------|-----------------------|----------------------|-----------------------|
| Low cost     |            |                       |                      |                       |
| High speed   |            |                       |                      |                       |
| Priorization |            |                       |                      |                       |

| Privacy      | Consent and notice | Use across clouds |
|--------------|---------------------|--------------------|
| Data protection |                     |                    |
| Data minimization |                   |                    |
| Privacy by design |                 |                    |

| Security     | Security by design | Coherent responsibility |
|--------------|--------------------|-------------------------|
| Accountability |                   |                         |

| Standards    | Open | Interoperability | Discoverability |
|--------------|------|-----------------|-----------------|
| Voluntary    |      |                 |                 |
| Encouraged by governments | | | |
| Open data, open access | | | |

| Ownership    | Incentivization | Ownership protection |
|--------------|----------------|----------------------|
security, because the latter one would threaten the current rate of technological and innovation advance (FTC Staff Report 2015). Instead, privacy-by-design and security-by-design strategies should be promoted to ensure that protection is embedded in the core design of the product, rigorously evaluated throughout its development process (Intel; FTC, ISOC) and that the volume of data manufacturers collect and maintain is minimised (FTC Staff Report 2015).

Although there are many companies whose strategic advantages depend on the implementation of closed and proprietary IoT systems, interoperability standardisation initiatives are critical for the successful development of IoT ecosystem. Standardisation efforts should be global, open and voluntary and encouraged by the governments. Such an approach will also increase adoption rate, because it will address the potential customers’ concerns on high ownership complexity and vendor lock-in. Open standards should be complemented by the open data and open access policies. In order to implement these policies, device, service and data discoverability is crucial. However, in order to incentivise the realisation of the opportunities provided by the ‘open’ policies, public policy must ensure protection of proprietary data and data ownership aspects, in general.

Domain framework for IoT systems implementation

The hypothesis of the work behind this paper was that requirements engineering, model-based systems engineering, IoT system architecture, maturity models, interoperability and policy and regulation aspects are all elements of the domain framework for IoT systems implementation.

In the discussion above, we have addressed recent research in the mentioned topics, in the context of IoT. The summary of identified challenges and opportunities is presented in Table 1. The table shows example technologies aimed to be candidates for meeting the designated opportunities.

Based on the identified opportunities, the proposed domain framework for IoT implementation is extended by considering more detailed overview of the technologies, approaches and specific aspects of each of the above identified elements. This extended view is shown in Figure 6.

The extended view identifies more specific relationships between domain elements and classifies cross-domain issues and opportunities. Following dependencies and cross-domain concepts as illustrated in Figure 6 has been found.

Effect of policy and regulatory aspects to the domain framework elements

Enhanced connectivity, namely new M2M services, will affect the structure of non-functional requirements collection and associated metadata. Access control schemas, used in MAS, must be considered at modelling level, and they will be formally expressed, while continuously taking into account data ownership issues, including licences of data use in distributed environment of IoT ecosystem. Similarly, in the development of model frameworks, both domain experts and system architects must follow privacy-by-design and security-by-design policies. Reliability of critical services, such as health care, safety and security, will be considered as non-functional requirements of highest priority and implemented in traffic prioritisation policies. Finally, capability to interoperate will be based on the open interoperability standards.

Core technical structure of the domain framework

The backbone of the framework consists of system requirements, modelling constructs that satisfy them and agents which implement the models. The centrepoint of this backbone is the model. It is either semantically annotated or formal model (so it facilitates inferring the meaning of the data coming from different sources); it is interpreted at run-time, by the implementation agent in MAS. Besides representing the agent environment, formal models are used to define goal and non-goal states, to be reached by the goal-based agents and utility functions to be used by the utility-based agents to measure how desirable perceived state is. In satisfying the system requirements and making sense of acquired data, besides defining structural (including data structures and restrictions, explicitly defined in domain ontology) and behavioural aspects (explicitly defined in application ontology), model also considers innovative views to the ecosystem, such as capabilities of its artefacts, maturity and trust.

The capability model is a cross-domain concept, which is proposed due to a need to abstract the heterogeneity of devices and to formally define the capability of one device or agent to interoperate with another. It is formally implemented as extension of W3C SSN ontology in application ontology.

The trust model is introduced by the need to facilitate the agent’s capability to acquire data from relevant, reliable and trustful sources. The trust model will also define formal requirements for models’ validation. Trust ontologies have started to emerge, even with specialisations in IoT domain (Taherian, Jalili and Amini 2008); most of the current models have been built on the O’Hara’s formal trust model of trustworthiness (O’Hara 2012).
The maturity model is a cross-domain concept, and it is used to formally define improvement path and maturity levels as agent goals. The maturity model is instantiation of the maturity assessment application ontology, which specialises the upper-level maturity assessment reference ontology in the domain (formally described by the domain ontology) and application context.

The application model is meta-model which is used to instantiate behavioural aspects of the IoT scenarios in the ecosystem, such as services (CRUD, processing, visualisation and others) and their orchestration (business processes), access control schema and user interfaces (if any).

Interoperability as foundation of open IoT ecosystem

Capability to interoperate is capability of agents in the IoT ecosystem to sense and interpret the meaning of acquired data and to make a decision to act upon this meaning, based on the formal models. RE methods must ensure that all system requirements consider the effect of their implementation on system interoperability. In order to facilitate trustful and reliable interoperation between two devices, this capability needs to be assessed by the agent, prior to interoperaton. Such assessment can take place by using interoperability maturity criteria and levels, formally defined in the maturity assessment application ontology.

Scenario

The relevance of the domain framework is shortly illustrated by the scenario of production planning and scheduling in shop floor.

Production planning aims at calculating the optimal set of variables, related to product and its bill of material, including lot size, quantities to make or buy, delivery date, starting and completion operation times and others. It is typically based on models which assume known and stable environment. Often, these models do not include the description of the system dynamics (e.g. resources state changes, failure occurrence and resources ageing) and the environment...
dynamics (e.g. changing demand mix and volume and cost of materials). In reality, good performance achieved by the system in determined conditions can drastically deteriorate if these conditions change.

One way to address this problem is smart automation in which the products, materials, tools, transport devices and other resources are able to take their own decisions concerning optimised production execution. IoT allows embedding more complex and more accurate data to inform these decisions.

In this scenario, a shop floor is a multi-agent system. Materials, parts and products are represented by goal-based agents, where the definition of goal for each of the items is based on the MRP and it is related to needed transformation (by cutting, drilling, milling, etc.) in a manufacturing process, which is determined based on the position of a resource in a bill of material and routing for the final product. Work centres are represented by utility-based agents, offering different transformation services (depending on tooling and configuration) to material, part and product agents, based on their respective capability models. Each of the latter can request and negotiate (based on their respective capabilities to interoperate) a specific transformation service from each of the work centre agents, where the success of this negotiation will depend on the required quality, capacity of the work centre and cost of reconfiguration needed for service provision. All physical stock moves (from physical stock locations in warehouses to a shop floor) are ensured automatically, after the successful negotiation between work centres, material or part and internal transport facilities. Thus, shop floor is on-demand system, because it does not assume pre-defined configuration of devices and fixed agreements on their communication; in contrast, it acknowledges a set of capabilities in a larger environment and relies on a formal reasoning to implement the specific interaction scenarios.

The shop-floor system is an IoT system, because: location sensor data are used to make the most optimal internal logistics decisions; accelerometers can be deployed for the purpose of predictive maintenance (vibration monitoring); ultrasonic sensors can be used to look for cavities in castings in a production line, etc. IoT system considers reliability of critical services, such as those related to work safety regulations (e.g. by prioritising data traffic from air quality sensors and smoke indicators) and ensuring safest internal logistics routes (proximity sensors).

Shop-floor IoT system is open in the sense that it is part of the larger ecosystem of suppliers, customers and service providers in a supply chain, whose access to data and IoT capabilities is restricted by access control schema (which implements trust model). This larger ecosystem is also explored by software sensors (agents) which track market information, relevant for MPS (material prices, seasonal factors, etc.). Selected RE method acknowledges the requirements of all stakeholders in this open ecosystem.

Shop-floor IoT system is formal model-driven because data are given the explicit meaning, formally defined in the vast number of existing domain and application ontologies. The domain framework foresees following key aspects of that model: domain, maturity, application, capability and trust. The latter four are domain-independent, and candidate formal models have been already mentioned above.

One of the strongest candidates for domain ontology is a collaborative production automation and control architecture (ADACOR) (Borgo and Leitao 2007). It formally describes the functions of manufacturing control system, such as process planning, scheduling and plan execution. It is highly convenient for formal modelling of resource-based organisation of shop-floor IoT ecosystem, because it is ‘built on a set of autonomous and cooperative holons, each one being a representation of a physical resource (CNC machine, robots, etc.) or a logic entity (orders, etc.).'

**Conclusion**

When looking at the past experiences in implementation of enterprise-wide IT systems, such as ERP systems, it is easy to assume that introducing more complexity to the scope of their operation will make the implementation problem even worse. Even though that the scientific community addressed ERP implementation problem at big scale, it is surprising to find out that this problem has not been considered so far in the context of IoT systems. Therefore, the motivation for the work presented in this paper was to establish the baseline for further research in this topic, based on the proposed domain framework for IoT systems implementation.

At the beginning, we have assumed that major challenges and opportunities related to implementation problem lay at several selected scientific domains and sub-domains. MBSE, RE and interoperability domains were quite obvious choices. While two former are universally relevant for IS implementation, the latter was a challenge related to unprecedented rate of heterogeneity in the environment where IoT systems are implemented. Maturity modelling was introduced because the success and impact of the devices interactions depend on the readiness of the organisation, people, systems and interacting agents to make sense of those interactions. Challenges related to policy and legal aspects affect all other domains, mostly due to the fact that boundaries of the IoT system, unlike traditional ERP systems, are not fixed anymore. Actually, their outreach is global, often spanning multiple jurisdictions.
Further analysis of the individual domains, in the context of IoT systems implementation, has produced detailed domain framework. It identified fine-grained challenges and technologies and approaches for their resolution, based on the literature review. It also highlighted new relationships between domains, hence, common research interests which, if addressed, could have multiplied impact to the research of IoT systems implementation problem.

The proposed domain framework can be used as practical checklist and blueprint for formal model-driven IoT ecosystem conceptualisation. However, it is not exhaustive and self-sufficient for the implementation process; it is not associated with well-defined methodology, which is a first priority for the future work in further development. Synthesis and alignment of the existing work in the development of capability, trust, maturity and application models will produce the basis for this methodology – integrated meta-model of the domain framework for implementation of IoT ecosystems.

**Disclosure statement**
No potential conflict of interest was reported by the authors.

**ORCID**

Alexis Aubry © http://orcid.org/0000-0002-7335-821X

**References**

Ahern, D. M., A. Clouse, and R. Turner. 2003. *CMMI Distilled: A Practical Introduction to Integrated Process Improvement*. Boston, MA: Addison-Wesley.

Ayala, I., M. Amor, L. Fuentes, and J. M. Troya. 2015. “A Software Product Line Process to Develop Agents for the IoT.” *Sensors* 15 (7): 15640–15660.

Becker, J., R. Knackstedt, and J. Pöppelbuß. 2009. “Developing Maturity Models for IT Management – A Procedure Model and its Application.” *Business & Information Systems Engineering (BISE)* 1 (3): 213–222.

Bello, E., F. Riguzzi, and S. Albani. 2011. “A Distribution Semantics for Probabilistic Ontologies.” In *Proceedings of the 7th International Workshop on Uncertainty Reasoning for the Semantic Web*, 75–86. Aachen: CEUR-WS.org.

Bencomo, N., G. Blair, and R. France. 2007. “Summary of the Workshop Models@run.time at MoDELS 2006.” *Models in Software Engineering*. Lecture Notes in Computer Science 4364, 227–231.

Borgo, S., and P. Leitao. 2007. “Foundations for a Core Ontology of Manufacturing.” In *Ontologies*, Volume 14 of the Series Integrated Series in Information Systems, 751–775. Boston, MA: Springer. doi: 10.1007/978-0-387-37022-4.27

Brazier, Frances M. T., B. Dunin Keplicz, N. R Jennings, and J. Treur. 1995. “Formal Specification of Multi-agent Systems.” Proceedings of the First International Conference on Multi-Agent Systems, ICMAS’95, 25–32. San Francisco, CA: MIT Press.

Browne, I. 2015. “Regulation and the Internet of Things.” Discussion paper. 15th Global Symposium for Regulators (GSR15). http://www.itu.int/en/ITU-D/Conferences/GSR/Documents/GSR2015/Discussion_papers_and_Presentations/GSR_DiscussionPaper_IoT.pdf.

Compton, M., P. Barnaghi, L. Bermudez, R. Garcia-Castro, O. Corcho, S. Cox, J. Graybeal, et al. 2012. “The SSN Ontology of the W3C Semantic Sensor Network Incubator Group.” *Web Semantics: Science, Services And Agents On The World Wide Web* 17: 25–32.

Costa, P. C. G., and K. B. Laskey. 2006. “PR-OWL: A Framework for Probabilistic Ontologies.” In *Proceedings of the 2006 Conference on Formal Ontology in Information Systems: Proceedings of the Fourth International Conference (FOIS 2006)*, 237–249. Amsterdam: IOS Press.

FTC Staff Report. 2015. Internet of Things, Privacy and Security in a Connected World. https://www.ftc.gov/system/files/documents/reports/federal-trade-commission-staff-report-november-2013-workshop-entitled-internet-things-privacy/150127iotrpt.pdf

Galy, E., and M. J. Sauceda. 2014. “Post-implementation Practices of ERP Systems and Their Relationship to Financial Performance.” *Information & Management* 51 (3): 310–319.

Garcia, C. G., C. P. G-Bustelo, J. P. Espada, and G. Cueva-Fernandez. 2014. “Midgar: Generation of Heterogeneous Objects Interconnecting Applications. A Domain Specific Language proposal for Internet of Things scenarios.” *Computer Networks* 64: 143–158.

Guédria, W., Y. Naudet, and D. Chen. 2015. “Maturity Model for Enterprise Interoperability.” *Enterprise Information Systems* 9 (1): 1–28.

Guinard, D., V. Trifa, S. Karnouskos, P. Spiess, and D. Savio. 2010. “Interacting with the SOA-based Internet of Things: Discovery, Query, Selection, and On-demand Provisioning of Web Services.” *IEEE Transactions on Services Computing* 3 (3): 223–235.

Halliday, J., and R. Lam. 2016. *Internet of Things. Some Legal and Regulatory Implications*. Baker&McKenzie. http://www.bakermckenzie.com/-/media/files/insight/publications/2016/02/internet-of-things_ar_australia_internetofthings_feb16.pdf?la=en.
Halper, Fern. 2016. TDWI IoT Readiness Guide, Interpreting your Assessment Score. https://tdwi.org/whitepapers/2016/08/tdwi-iot-readiness-guide.aspx.

Harrand, N., F. Fleurey, B. Morin, and K. Elif Husa. 2016. “ThingML: A Language and Code Generation Framework for Heterogeneous Targets.” Proceedings of the ACM/IEEE 19th International Conference on Model Driven Engineering Languages and Systems, 125–135. New York: ACM.

Herding, R., and L. Mönch. 2016. “S2CMAS: An Agent-based System for Planning and Control in Semiconductor Supply Chains.” Proceedings of Multiagent System Technologies: 14th German Conference, MATES 2016, 115–130. Cham: Springer International Publishing.

HP. 2014. HP Study Reveals 70 Percent of Internet of Things Devices Vulnerable to Attack. http://www8.hp.com/us/en/hp-news/press-release.html?id=1744676#/WG1xxRsrKUl.

IDC. 2013. Business Strategy: The Coming of Age of the “Internet of Things” in Government, IDC.

Intel. 2014. Policy Framework for the Internet of Things (IoT), Intel report. https://www.intel.com/content/www/us/en/policy/policy-iot-framework.html.

ISO. 2016. The Internet of Things, An Internet Society Public Policy Briefing. http://www.internetsociety.org/policybriefs/iot

Katasonov, A., O. Kaykova, O. Khriyenko, S. Nikitin, and V. Terziyan 2008. “Smart Semantic Middleware for the Internet of Things.” Proceedings of the 5th International Conference on Informatics in Control, Automation and Robotics – Volume 1: ICINCO, 169–178. Setúbal: ScitePress. doi: 10.5220/0001489001690178.

Kohlegger, M., R. Maier, and S. Thalmann. 2009. “Understanding Maturity Models Results of a structured Content Analysis.” InKNOW ’09 and I-SEMANTICS ’09, Graz, Austria.

Kotonya, G., and I. Sommerville. 2002. Requirements Engineering: Process and Techniques. New York: John Wiley and Sons.

Lahrmann, G., F. Marx, and T. Mettler, et al. 2011. “Inductive Design of Maturity Models: Applying the Rasch Algorithm for Design Science Research.” International Conference on Design Science Research in Information Systems. Berlin, Heidelberg: Springer, 176–191.

Leffingwell, D. 2011. Agile Software Requirements: Lean Requirements Practices for Teams, Programs, and Enterprise. Boston, MA: Addison-Wesley. ISBN-10: 0321635841.

Lichtblau, K., V. Stich, R. Berenrath, M. Blum, M. Bleider, A. Millack, K. Schmitt, E. Schmitz, and M. Schröter. 2015. IMPULS – Industrie 4.0- Readiness. Aachen-Köln: Impuls-Stiftung des VDMA.

Madu, C. N. 1998. Handbook of Total Quality Management. Boston, MA: Springer.

Manate, B., V. I. Munteanu, and T. F. Fortis. 2013. “Towards a Scalable Multi-agent Architecture for Managing IoT Data.” Proceedings of 2013 Eighth International Conference on P2P, Parallel, Grid, Cloud and Internet Computing (3PGCIC 2013), 270–275. New Jersey: IEEE. doi: 10.1109/3PGCIC.2013.46

Manyika, J., M. Chui, J. Bughin, R. Dobbs, P. Bisson, and A. Marrs. 2013. Disruptive Technologies: Advances that Will Transform Life, Business, and the Global Economy. Report McKinsey Global Institute. https://www.mckinsey.com/business-functions/digital-mckinsey/our-insights/disruptive-technologies.

McCarthy, J., and S. Buvac. 1997. “Formalizing Context (Expanded Notes)”. http://philpapers.org/rec/MCCFCE.

McKinsey report. 2012. “Delivering Large-scale IT Projects on Time, on Budget, and on Value.”

McUmber, W. E., and B. H. Cheng. 2001. “A General Framework for Formalizing UML with Formal Languages,” Proceedings of the 23rd International Conference on Software Engineering. Washington, DC: IEEE Computer Society, 433–442.

Mettler, T. 2011. “Maturity Assessment Models: A Design Science Research Approach.” International Journal of Society Systems Science 3 (1/2): 81–98.

Molina, A., H. Panetto, D. Chen, L. Whitman, V. Chapurlat, and F. Vernadat. 2007. “Enterprise Integration and Networking: challenges and trends.” Studies in Informatics and Control, Informatics and Control Publications. 16 (4): 353–368.

O’Hara, K. 2012. A General Definition of Trust. Technical report, University of Southampton.

Paredis, C. 2011. Model-based Systems Engineering: A Roadmap for Academic Research. Atlanta, GA: Frontiers in Model-based Systems Engineering.

Patel, P., and D. Cassou. 2015. “Enabling High-level Application Development for the Internet of Things.” The Journal of Systems and Software 103 (2015): 62–84.

Paulk, M. 1993. “Capability maturity model for software.” In Encyclopedia of Software Engineering, Volume 2, edited by J. J. Marciniak, 928–929. New York: John Wiley & Sons.

Rivera, D., L. Cruz-Piris, G. Lopez-Civera, E. de la Hoz, and I. Marsa-Maestre. 2015. “Applying an Unified Access Control for IoT-based Intelligent Agent Systems.” IEEE 8th International Conference on Service-oriented Computing and Applications (SOCA). New Jersey: IEEE. doi: 10.1109/3PGCIC.2013.46

Rockwell Automation. 2014. “The Connected Enterprise Maturity Model.”

Schumacher, A., S. Erol, and W. Sihn. 2016. A Maturity Model for Assessing Industry 4.0 Readiness and Maturity of Manufacturing Enterprises. Procedia CIRP 52, 161–166.
Schwartz, T., I. Zinnikus, H.-U. Krieger, C. Bürckert, J. Folz, B. Kiefer, P. Hevesi, et al. 2016. “Hybrid Teams: Flexible Collaboration Between Humans, Robots and Virtual Agents.” Proceedings of Multiagent System Technologies: 14th German Conference, MATES 2016, 131–146. Cham: Springer International Publishing. doi: 10.1007/978-3-319-45889-2_10.

Song, Z., A. A. Cardenas, and R. Masuoka. 2010. “Semantic Middleware for the Internet of Things.” Internet of Things (IOT) Conference, 1–8. New Jersey: IEEE. doi: 10.1109/IOT.2010.5678448.

Sriram, R. D., and A. Sheth. 2015. “Internet of Things Perspectives.” IT Professional 17 (3): 60–63.

Taherian, M., R. Jalili, and M. Amini. 2008. PTO: A Trust Ontology for Pervasive Environments. 22nd International Conference on Advanced Information Networking and Applications – Workshops, Okinawa, 301–306.

Tarhan, A., O. Turetken, and F. Van Den Biggelaar. 2015. “Assessing Healthcare Process Maturity: Challenges of Using a Business Process Maturity Model.” 9th International Conference on Pervasive Computing Technologies for Healthcare (PervasiveHealth), 339–342. New Jersey: IEEE. doi: 10.4108/icst.pervasivehealth.2015.259105.

Vachterytė, V. 2016. “Towards an Integrated IoT Capability Maturity Model.” BS thesis. University of Twente.

Verma, P., M. Gupta, T. Bhattacharya, and P. K. Das. 2014. “Improving Services using Mobile Agents-based IoT in a Smart City.” Review of. 2014 International Conference on Contemporary Computing and Informatics (Ic3i), 107–111.

Vernadat, F. B. 1996. Enterprise Modeling and Integration: Principles and Applications. London: Chapman & Hall.

Wang, W., S. De, R. Toenjes, E. Reetz, and K. Moessner. 2012. “A Comprehensive Ontology for Knowledge Representation in the Internet of Things.” Proceedings of TrustCom, Liverpool, UK, 1793–1798.

Weyns, D. 2010. Architecture-based Design of Multi-agent Systems. Berlin: Springer Science & Business Media. doi: 10.1007/978-3-642-01064-4.

Wooldridge, M., N. R. Jennings, and D. Kinny. 2000. “The Gaia Methodology for Agent-oriented Analysis and Design.” Autonomous Agents and Multi-Agent Systems 3 (3): 285–312.

Zambonelli, F. 2016. Towards a General Software Engineering Methodology for the Internet of Things. Computing Research Repository (CoRR). http://arxiv.org:443/find/cs/1/au:+Zambonelli_F/0/1/0/all/0/1

Zdravković, J., J. Stirna, J. C. Kuhr, and H. Koç. 2014. “Requirements Engineering for Capability Driven Development.” Proceedings of 7th Working Conference on the Practice of Enterprise Modeling (PoEM 2014), Springer LNBIP Vol. 197, 193–207.

Zdravković, M., F. Luís-Ferreira, R. Jardim-Goncalves, and M. Trajanović. 2017. “On the Formal Definition of the Systems’ Interoperability Capability: An Anthropomorphic Approach.” Enterprise Information Systems. 11 (3): 389–413.