Technology Management through Architecture Reference Models: A Smart Metering Case

Matus Korman, Robert Lagerström, Margus Välja, Mathias Ekstedt, Rikard Blom
KTH Royal Institute of Technology, Stockholm, Sweden

Abstract--**Enterprise architecture (EA) has become an essential part of managing technology in large enterprises. These days, automated analysis of EA is gaining increased attention. That is, using models of business and technology combined in order to analyze aspects such as cyber security, complexity, cost, performance, and availability. However, gathering all information needed and creating models for such analysis is a demanding and costly task. To lower the efforts needed a number of approaches have been proposed, the most common are automatic data collection and reference models. However these approaches are all still very immature and not efficient enough for the discipline, especially when it comes to using the models for analysis and not only for documentation and communication purposes. In this paper we propose a format for representing reference models focusing on analysis. The format is tested with a case in a large European project focusing on security in advanced metering infrastructure. Thus we have, based on the format, created a reference model for smart metering architecture and cyber security analysis. On a theoretical level we discuss the potential impact such a reference model can have.**

I. INTRODUCTION

Modeling and analysis of enterprise and system architectures provide capabilities that aid effective management of technology [22], [29]. Examples include the capability to hold an architectural overview of an enterprise and its IT environment (with support of as-is models) [31], to evaluate the viability of prospective improved scenarios (using to-be models), and to analyze different attributes within the architectures represented by models (e.g., interoperability [48], modifiability [28], flexibility [33], availability [14] and cyber security [19], [46]). However, both modeling and analysis of architectures have mostly been done using human effort. Today, approaches advancing the modeling and the analysis from mostly manual to largely automated [18], [49] respectively [24], [25], [31], [35] are receiving more attention. A vision of such approaches is to foster the development of EA tools that would essentially correspond to tools for computer aided design (CAD) used in other domains of engineering. The aim is to enable automated analysis and simulation of EA scenarios in a similar fashion as a whole spectrum of physics aspects can be analyzed and simulated in today's CAD tools. Thus, the effort is spent before the modeled system is actually built and making it less costly to make changes.

However, gathering facts about architectures and creating the appropriate models remains a lengthy and demanding process. Let us consider three model cases.

1) As-is modeling of large IT environments. The process of gathering data about and modeling IT environments hosting large enterprise systems (e.g., SAP) or large industrial control systems (e.g., ABB Network Manager) together with a myriad smaller systems they interface, might easily take months of intensive work. Hence, the data collection and modeling process in itself is an expensive and demanding part of architectural work, which calls for support.
2) Automated EA analysis of to-be scenarios. An analytical metamodel (e.g., [19]) might require or notably benefit from having models with high amount of detail as input. At the same time however, analysts and organizations creating such to-be models might be hindered by the large amount of uncertainty present in the process. For instance, consider an industrial plant evaluating a migration to a new control system. Obtaining exact and detailed data would be difficult, since the organization does not yet possess an implementation of such system, nor has its detailed documentation available. Consequently, the analyst is forced to make numerous assumptions, which can require tedious intelligence gathering, consulting, or plain guessing.
3) New technological trends. Innovation that continuously takes place around the globe yields new types of solutions that offer improvement potential for organizations, but at different expenses and risks. Hence, there is a need to evaluate these before making an investment decision. Examples may include cloud computing [32], the 3rd platform [20], IoT [4], or software-defined networking [42]. While some might still be defined very loosely, others can be modeled with more exactitude and detail. Moreover, models of these architectures can be reused over time and even between specific implementations.

To sum up, modeling for EA analysis often requires a lot of details. In addition to the fact that EA data collection is an expensive process as such. The cost of collecting data can be reduced through prioritizing what data to collect [37], automatic data collection [18], [49], and reuse of models including using so called reference models (as proposed in this paper).

While a number of contributions have been done in the field of reference architectures, an extension of the concept for purposes of automated EA analysis has not been proposed. In this article, we present a format for reference models designed for enabling reuse in modeling for

automated EA analysis.

The reference model format is suited for object models following the syntax of UML [38] class models. The format allows the use of a programming language such as OCL [39], and allows representing uncertainty through the extensions of P2AMF [24], [25]. We also argue that the use of reference models in the proposed format can decrease the modeling effort needed to model large IT architectures. We discuss this using a case study as a proof of concept. The case study is based on a large European project on security in advanced metering infrastructure. We have, based on the proposed format, created a reference architecture for smart metering systems and cyber security analysis of such.

This article unfolds as follows. Chapter II introduces the concept of reference architecture as used in literature and presents related work. Chapter III introduces the issue of reuse, and lists a few attributes relevant for reusing EA models. Chapter IV introduces the concept of EA analysis, and a few EA analysis frameworks. Chapter V describes the design of this study. Chapter VI describes the context of using reference models as employed in this study and chapter VII presents our proposed format of reference models for EA analysis. Subsequently, chapter VIII evaluates the reference model format using a real case on advanced metering systems. Chapter IX provides a discussion, directions for future work, and finally chapter X concludes the study.

II. THE CONCEPT OF REFERENCE ARCHITECTURES AND RELATED WORK

The purpose of this section is twofold. First, it introduces the concept of reference architectures – something that a reference model describes. Second, it briefly mentions contributions that relate to this study and reference architectures in a broader sense.

Reference architectures have been proposed for a number of domains [1], [20], [32] and even a framework for classifying reference architectures has been proposed [1].

Bernus & Noran [6] differentiate between two types of architectural models commonly proposed by EA scholars. The first, type 1 architectures, also called reference models or partial models, have the essence of generally applicable blueprints. The second, type 2 architectures, also called enterprise reference architectures, have the essence of life cycle architectures. In GERAM [5] (generalized enterprise reference architecture and methodology), the two types of architectures correspond to the concepts of PEM (partial enterprise model) and GERA (generalized enterprise reference architecture), respectively. In literature however, the term reference architecture often refers to a type 1 architecture (reference model).

Cloutier et al. [10] provide a comprehensive and systematic overview of the concept of reference architecture. In spite of varying specific understanding of the concept across scientific literature and practice, reference architecture is here seen as an artifact with the purpose to provide guidance for further developments (e.g., through facilitating multi-site, multi-vendor, multi- [...] system creation and life-cycle support, effective creation of new products, and achieving interoperability between many different and evolving systems). Further according to that view, a reference architecture usually captures the accumulated architectural knowledge of thousands of man-years of work, ranging from *why* (e.g., value chain, application, etc.), *what* (e.g., systems, functions, etc.), to *how* (e.g., design views and diagrams, design patterns, etc.). According to [10], objectives of reference architectures include managing synergy, providing guidance in form of architecture principles and best practices, an architecture baseline and an architecture blueprint, capturing and sharing architectural patterns, providing a common lexicon and taxonomy, a common architectural vision, modularization and the complementary context, articulation of domain and realization concepts, explicit modeling of functions and qualities above systems level, and explicit decisions about compatibility, upgrade and interchangeability.

Several studies attempt to provide general models of reference architectures or guidance in design of reference architectures. Examples include Nakagawa et al. [36] for EA reference architectures, Galster & Avgeriou [15] for software architectures, and Irlbeck et al. [21] for smart energy systems.

There are also proposals of formats for EA reference models [9], [21]. However, none of the identified works propose a reference model format explicitly and generically suited for automated EA analysis [14], [24], [25], [48]. The closest match in terms of included detail and thus support to automated analysis is the use of reference models in software engineering [34].

Since creating models suitable for automated EA analysis poses increased demands for detailed data (e.g., costs, failure rates, probabilities of events, configuration details and other properties) and so becomes more costly, increasing the modeling efficiency helps keeping the whole EA analysis process more viable.

Using reference models, which contain both reusable structure and generic assumptions regarding detailed data, brings reuse into modeling in a way that can satisfy the needs of metamodels for automated EA analysis.

III. REUSE AS THE PURPOSE OF REFERENCE MODELS

The nature and the use of reference models indicate its main purpose – to promote reuse of architectural knowledge – be it in a context of industrial development, analysis, research, education, or some other.

The issue of reuse has been around for about two decades, mostly in the domain of software engineering. Frakes & Kang [13] provide a brief overview of software reuse research,

among other mentioning domain engineering (also known as product line engineering) and a number of methods and techniques therein, which attempt to promote software productivity through reuse. Somewhat closer to the scope of reference models, Robinson et al. [44] provide their views on the issue of reuse of simulation models (i.e., artifacts ranging from simple mathematical models to software systems for military simulations). Although these works relate to the subject of this study, their approaches to supporting the reuse of architectural knowledge are often very different from those applicable to reference models of enterprise architectures.

Further toward the issue of model reuse, Robinson et al. [44] identifies two major influencing factors: (1) the *validity and credibility* of the model; and (2) the *costs and benefits* of reuse, which undoubtedly also apply to EA models. The work also mentions a major obstacle to reuse – the *composability problem* [of the reused model with its environment]. The composability problem also applies to EA models, since EA models typically follow some predefined syntax (e.g., [41]), and reusing a model with entirely different syntax might imply the need of a major rework. In EA analysis, the need to additionally capture a multitude of properties of modeled entities according to an analytical metamodel even rises the challenge. Yet another piece of EA-applicable remark in the work relates to the pitfalls of reusing models – the *abstraction challenge*, saying that the simplest model fitting the purpose is typically the best. When reusing a model, the level of abstraction might not be readily compatible with the rest of the model environment, which it shall enrich, which leads to the risk that an overly detailed model would still be employed in an analytic task for the sake of reuse.

Finally, Pidd [43] lists four properties that a reuse strategy should support (related to models for simulation): (1) *abstraction* as the ability to provide succinct, high-level descriptions of reuse artifacts to ease the understanding of their purpose, nature and behavior; (2) *selection* to aid performing reuse through mechanisms that allow the location, comparison and selection of reuse artifacts; (3) *specialization* (modification) of the reuse artifacts into usable, concrete entities; and (4) *integration* to ease the combination, connection and communication between reuse artifacts. All of the above likely apply to reference models in EA as much as to models for simulation or pieces of software reuse artifacts.

IV. BACKGROUND

Enterprise architecture (EA) is defined as "*a coherent whole of principles, methods, and models that are used in the design and realization of an enterprise's organizational structure, business processes, information systems, and infrastructure*;" it attempts to capture the essentials of the business, IT and its evolution [31]. In this light, EA can be adopted and used for a multitude of purposes including documentation, communication support, design, analysis, transformation, decision-making et cetera [29]. In this paper we use the term enterprise architecture, but ofte we also mean system architecture or especially system-of-systems architecture (like in our AMI case in chapter VIII). The common denominator is that the models of these architectures are often holistic, large, complex, contain many different types of objects, and usually requires more than human effort to be really useful in analysis and simulation.

Metamodeling [2], [27] can be briefly described as "*the modeling of models*" [47]. A meta- model describes the syntax of its instance models and their permitted structure (i.e., classes with their properties, associations with their cardinality constraints). A corresponding instance model can then describe instances of the classes (i.e., objects with their property values), and instances of the associations (i.e., connections).

Similarly as a metamodel relates to instance models, does a meta-metamodel relate to meta-models. An example of such meta-metamodel is the Meta Object Facility (MOF) [40]. Recent contributions have refined this line of abstraction even further through the concepts of clabject and power type, into level-agnostic modeling [3], [16], providing additional freedom of abstraction as compared to MOF. Although metamodeling is commonly seen as a means of specifying and constraining models on a lower level of abstraction, metamodels can also possess active analytic capabilities (e.g., by embedding executable code for the purposes of inference).

In recent years, EA analysis has been receiving more attention [24], [25], [30], fueled by decision makers' needs to compare different alternative scenarios and make trade-offs between their different aspects (e.g., service availability [14], interoperability [48], cyber security [19], [46], or costs [11],[23],[26]) to predict the effects of prospective decisions. Rather than as a purely human intellectual activity, this study views the concept of EA analysis as a process largely automated through computational processing that evaluates arbitrary aspects of EA models. Lankhorst [31] identified two dimensions for classifying techniques of EA analysis: (1) functional vs. quantitative; and (2) analytical vs. simulation. While functional analyses can be used to gain insight into the behavior of an architecture, identify impacts of changes to it, or validate its correctness; quantitative analyses can provide quantitative answers such as "how much downtime" etc. While analytical techniques typically calculate a unique, reproducible result; simulation techniques "run" the model and can employ stochastic techniques.

A. Framework for enterprise architecture analysis

The Predictive, Probabilistic Architecture Modeling Framework (P2AMF) [24], [25], is a framework enabling automated analysis of architectures, based on metamodels. Although primarily intended for quantitative simulations, it can also employ analytical techniques, mostly quantitative.

P2AMF extends the combination of Unified Modeling Language (UML) [38] and Object Constraint Language (OCL) [39] in three major ways. First, it allows one use stochastic value expressions by using probability distributions (e.g., "normal(25, 5)" for an input ($\in$ R) from normal distribution with $\mu = 25$ and $\sigma 2 = 5$, or "bernoulli(0.75)" for a boolean input with 75% chance of yielding true). Second, it enables specifying stochastic existence of objects (i.e., structural uncertainty). Third, it uses Monte Carlo approach for simulation: the model is first sampled into a number of deterministic UML/OCL diagrams, which are subsequently evaluated according to the OCL logic, UML structure, and objects' property values.

B. Analytical metamodels

There are numerous metamodels available for architecture analysis, in this subsection we mention a few. The reason we for instance choose to briefly present CySeMoL below is that it is a predecessor of a commercially available modeling tool for architecture modeling and cyber security analysis called securiCAD[1] [12].

The Cyber Security Modeling Language (CySeMoL) [19], [46] is a metamodel for the analysis of cyber security in IT architectures, implemented in the P2AMF framework [24], [25]. CySeMoL strives for a holistic approach to evaluating security and is even built for coping with uncertainty. It defines over twenty assets and over hundred different attack steps and defense mechanisms.

CySeMoL uses Bayesian networks and attack graphs [45] to produce a map of reachability across an IT architecture model. The resulting map contains a set of probabilities, with which the attacker, given certain assumptions, can reach each attack step of each asset present in the model.

Other similar metamodels include a framework for analysis of IT service availability [14], a metamodel for interoperability analysis [48], and an educational-grade metamodel for multi- attribute prediction [11],[23],[26].

As a commonality among the above-mentioned metamodels, each entity (e.g., an operating system or an IT service) commonly has a set of properties, the values of which would usually occur within certain ranges and follow certain probabilistic distributions for a specific piece of reality modeled (for some examples, Fig.s 6 & 7). Hence, there is a potential for generalizing and subsequently reusing parts of the models. The reuse can apply within a single model, across multiple models, and even across organizations and modeling efforts taking place at different times.

V. STUDY DESIGN

The study builds on the foundations of design science [17], proposes a reference model format suited for reuse in modeling for EA analysis, and evaluates its contributory potential on a real case. Below, this section provides a brief summary of the study.

Research in automated EA analysis has gained attention, and a number of analytical metamodels have already been developed. At the same time, reference models do not typically contain information needed by analytical metamodels, and neither has a format for machine-friendly representation of such information in reference models been proposed. Hence, the need for the format presently exists, and is seen as the primary problem within this study. Additionally, as researchers and practitioners face greater needs for improving the efficiency of and reuse in modeling for automated EA analysis, custom formats of reference models might start being proposed, likely incompatible with one another. Considering the difficulties such a diversification might pose for the interoperability of solutions for EA modeling and analysis in the long run, the need to take an early step toward unification is a secondary problem within the study.

To achieve a highly interoperable and hence easily adoptable artifact, the research process was constrained by adherence to the bases of UML [38], and used the technological bases of OCL [39] and P2AMF [24], [25]. The research process led further through existing analytical metamodels [11], [14], [19], [23], [26], [46], [48], specifically in the reliance on sets of attributes found to notably impact the level of an attribute under consideration (e.g., cyber security or service availability).

The study aimed at formulating a format that is interoperable, expressive, and easy to use, to allow reducing modeling effort, provided that a generic reference model can be reused in the modeling process. Requirements for the format were on the one hand the full backward compatibility with UML and OCL, and on the other hand the possibility to represent uncertainty to more generically enable reuse in modeling for EA analysis.

The artifact resulting from this study can be seen as an enrichment or an extension of the concept of partial enterprise models (PEMs) as presented in GERAM [5], by the ability to stochastically express uncertainty of data values and object-relational structure, and hence improved support for model reuse.

The evaluation of the format takes into account a model of a complex IT environment, which is quantitatively analyzed according to the difference in modeling effort needed, both in an ideal and an assumed realistic scenario of application. The modeling effort is measured by counts of objects and connections to instantiate, and counts of property values to set or adjust.

The contribution of this study is a piece of design construction knowledge (i.e., foundations [17]) regarding a format for representing reference models that may contain

[1] For more information about securiCAD visit: www.foreseeti.com

uncertainty, enabling greater reuse in models for automated EA analysis.

Although this article is mostly intended for a technical audience familiar with EA (e.g., EA solution architects, analysts and scholars) and modeling technologies (especially UML and OCL), the article also contains useful information for managers involved in EA decision-making.

VI. THE USE OF REFERENCE MODELS IN EA ANALYSIS

This chapter tries to describe the use of reference models in EA in terms of a workflow, and so provide frames for better understanding of the concept of reference model as treated in this study.

First of all, we would like to note that a model is an excessively broad category of artifacts, which can be divided into concrete models (e.g., mechanical or software models), as well as abstract ones (e.g., mathematical models, mental models, or architectural models). Further in that direction, an UML-like object-relational model is merely a subset of architectural models, which delimits this study, and the concept of reference model as treated herein.

Considering a simple modeling work flow (illustrated in Fig. 1), in which a person uses a computer to create, maintain and draw benefits of EA models, as well as through our own modeling experience, we identified a set of requirements for the functionality that could aid EA modelers and other users in performing their tasks related to EA models, besides just the information present in them as in static documents.

The requirements are as follows:

- The availability of a reference model shall offer the potential for the modeler to either reuse modeling content and so save modeling effort, or reuse otherwise unavailable knowledge, as compared to a situation without the reference model;
- A reference model must be able to express a snapshot of a piece of concrete architecture;
- A reference model shall be able to express uncertainty in case several architectural alternatives are common for the architecture under consideration (e.g., several values or a range instead of just a single value; or several structural alternatives instead of just one structural alternative);
- A reference model shall be able to express validation constraints, so as to enable a software tool issue warnings to the modeler upon omitting important aspects of what needs to be modeled for correct representation, or breaching constraints that need to be complied to with regards to a given domain and the modeled architecture;
- A reference model shall be able to express logic for suggesting default alternatives, best choices, or best guesses when applicable;
- A reference model shall be possible to further specify and arbitrarily modify according to the modeling needs.

Clearly, the format of reference models itself cannot provide all the above-mentioned functionality, a modeling tool as a piece of software can (which is beyond the scope of this study).

However, the reference models complying with the format must be able to contain the above-mentioned information for the functionality to be provided by the tool, in connection with each specific reference model.

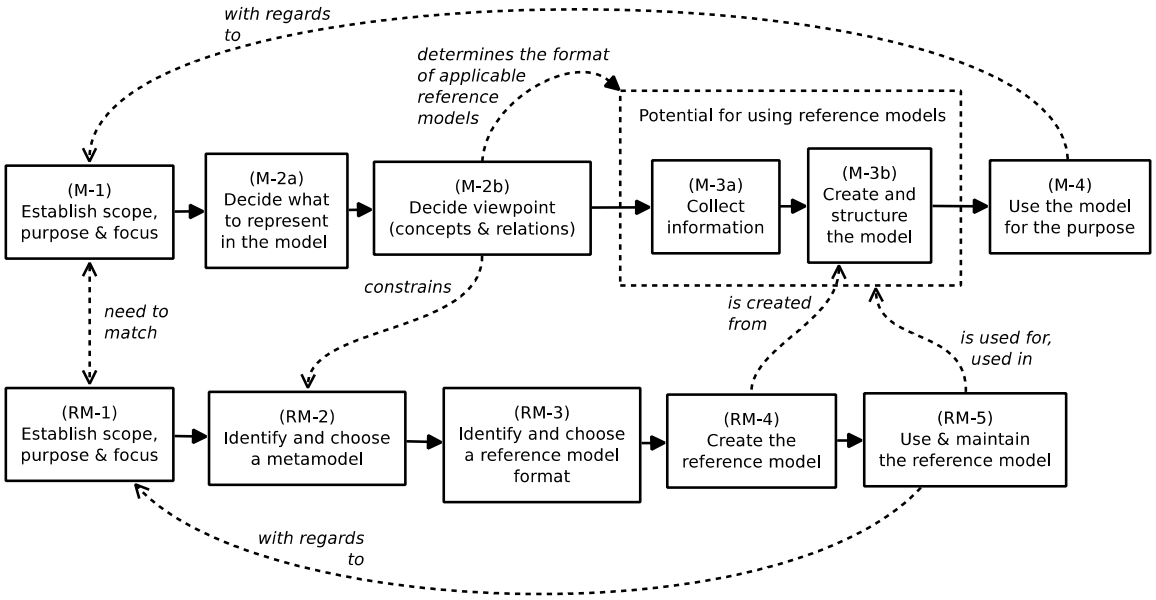

Fig. 1. Illustration of a modeling workflow: the upper row denotes the handling of a larger EA model, while the lower row denotes handling of a reference model, which is potentially used in the former.

VII. FORMAT OF REFERENCE MODELS FOR EA ANALYSIS

The reference model format reflects the MOF [40] meta-metamodel, besides some slight differences. This study focuses on the concepts; *metamodel*, *instance model*, and *template*, the last of which is intended to contain a reference model, according to the following definitions (using set notation) and as seen in Fig. 2:

Metamodels $\mathbf{M} \subset (C, A, I)$
- Classes $C \subset (name, T)$: $name \in$ String
- Attributes $T \subset (name, type, defaultDerivation)$:
  - $name \in$ String,
  - $type \in$ DataType,
  - $defaultDerivation \in$ String
- Associations $A \subset (cls_1, cls_2, name, cardmin, cardmax)$:
  - $cls_1 \in C$, $cls_2 \in C$,
  - $name \in$ String,
  - $cardmin \in$ N, $cardmax \in$ N
- Invariants $I \subset$ (class, *name, derivation*)
  - $class \in$ C,
  - $name \in$ String,
  - $derivation \in$ String

Instance models I $\subset$ (m, *O, R*) : $m \in$ M
- Objects $O \subset (class, P, name, existenceProbability)$:
  - $class \in C$,
  - $name \in$ String,
  - $existenceProbability \in$ R[0,1]
- Properties $P \subset (object, attribute, derivation)$:
  - $object \in O$,
  - $attribute \in T$,
  - $derivation \in$ String
- Connections $R \subset (obj_1, obj_2, association, existenceProbability)$:
  - $obj_1 \in O$, $obj_2 \in O$,
  - $association \in A$,
  - $existenceProbability \in R^{[0,1]}$

Templates $\mathbf{T} \subset$ (m, *O, R*) : $m \in \mathbf{M}$

(Immediately after instantiation of $t \in \mathbf{T}$ in $i \in \mathbf{I}$):

$m \in \mathbf{M}$, $i \in \mathbf{I}$, $t \in \mathbf{T}$, $i \in$ N : $t_i \subseteq i \Rightarrow mt_i \subseteq mi$

This study treats the concepts as follows. A metamodel consists of a set of classes, associations and invariants. A class contains its name and a set of attributes (cf. property in MOF). An attribute consists of its name, its data type, and its default derivation (in a programming language such as OCL), which can in its simplest form express the default value of all corresponding properties (property here seen as an instance model level entity corresponding to an attribute on the metamodel level). An association is identified by a directed relation between two classes together with its name, and further contains its minimal and maximal cardinality. An invariant consists of a name, and a derivation, which returns a Boolean value, based on whether the constraint it expresses is satisfied, or not. An instance model consists of a metamodel, to which it corresponds, a set of objects and a set of connections between the objects. An object is identified by its class, its properties (corresponding to the class' attributes), its name, and the probability of its existence. A connection is a directed relation between two objects, corresponding to an association (defined in the metamodel), and further having a probability of its existence. A template is in fact a special type of instance model; they are similar in essence. Both always correspond to a given metamodel. In order for a template to be instantiable in an instance model, the metamodels of the former has to be equal or at least be a subset of the latter. Provided this criterion, multiple templates can be instantiated in an instance model, and a single template can be instantiated multiple times.

For the sake of interoperability, the reference model format is built on the foundations of UML [38]. Since the format needs to define computation between values of properties and on the level of associations, there is a need for a expressions of a programming language used within the models (e.g., OCL [39]). UML class diagrams define the structure of models, their class/object-relational syntax, and how data (e.g., parameters in form of properties) are defined and stored in the models. A programming language such as OCL enables in-model computation (e.g., derivation of property values for the purposes of inference) and elegant querying for data within the object-relational structure of a model.

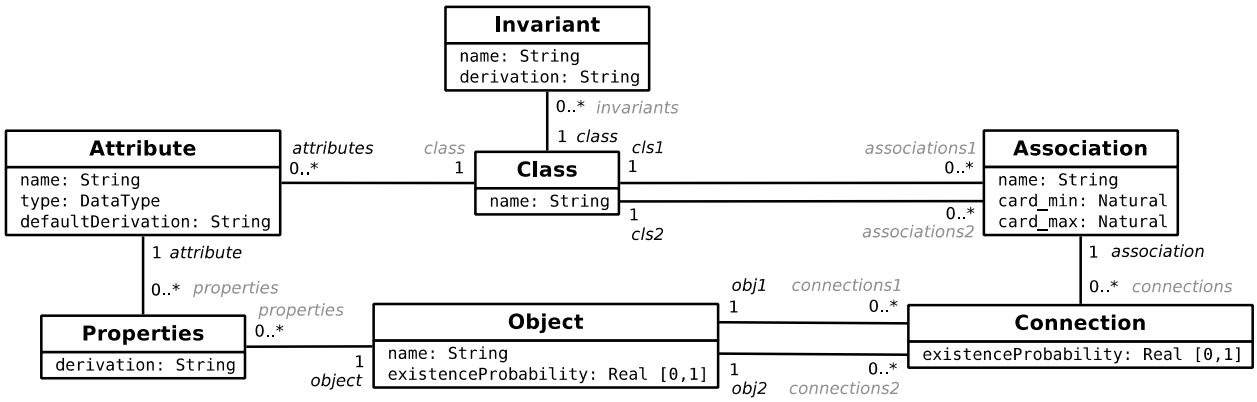

Fig. 2. UML depiction of the proposed reference model format.

The ability to represent uncertainty is a major feature of the proposed reference model format, which opens for greater reuse in modeling for EA analysis. Uncertainty tends to arise in the process of abstracting from very specific, real models, into generic models intended for broader reuse. For instance, a concrete installation of an operating system might be configured highly specialized while the configuration of a typical installation of that operating system is different, more generic, and some parameters fall within ranges rather than exact, point-like, quantities. Hence, in addition to exact quantities and structure, a reference model can express ranges with probability distributions of quantities, as well as multiple structural alternatives. The proposed format suggests the use of P^{2}AMF [24], [25], which supports expressing stochastic uncertainty in models, and at the same time fully preserves the backward compatibility with both UML and OCL.

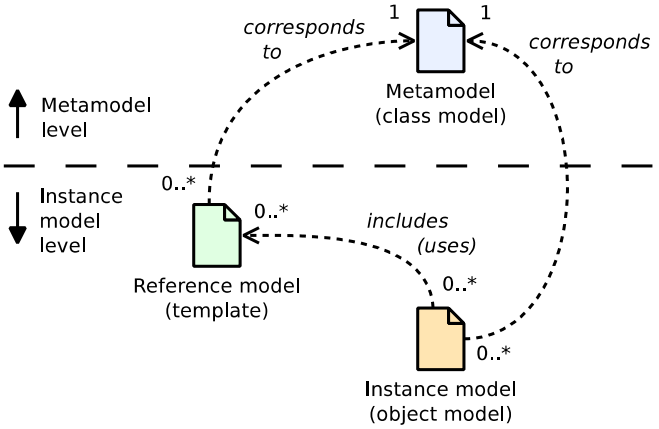

Fig. 3. Relationships between a reference model, metamodel and instance model.

From a persistence perspective, templates containing reference models are meant to reside independently of instance models, while their content is truly instantiated in an instance model first upon its use (inclusion), as depicted in Fig. 3.

The use of a reference model consists in the event of instantiating its content in an instance model (see Fig. 4), much like by running a script there. Hence, a reference model can be used in an instance model multiple times, and each of the sets of elements instantiated by the use of the reference model can afterwards be arbitrarily modified in the instance model, without it affecting the original reference model nor other instantiations of it. This flexibility renders useful when applied to similar yet not equal parts of an instance model.

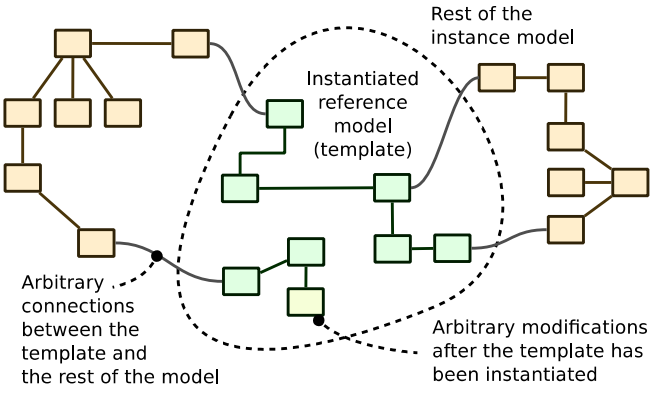

Fig. 4. Reference model inside an instance model (illustration).

An example application of reference models is illustrated in chapter VIII.

VIII. CASE: CYBER SECURITY IN AMI SYSTEMS

The demand for energy is increasing at the same time as we need to decrease our CO_2 emissions. In 2007 the European Union agreed upon a set of goals in order to reduce CO_2 emissions and energy consumption, as well as raising the shares of green power and the energy efficiency. More precisely the goals for 2020 are: 20% cut in greenhouse gas emission (compared to 1990 levels), 20% of EU energy from renewables, and 20% improvement in energy efficiency. This is called the 20-20-20 targets.

In order to reach the energy efficiency target the European Union has adopted a number of measures, incl. the planned rollout of close to 200 million smart meters for electricity and 45 million for gas by 2020. The result of this rollout alone is calculated to reduce emissions by 9% and an annual household energy reduction by comparable amounts. When a smart device is going to be delivered to almost 80% of the households in Europe there is lot of things that needs to be considered in terms of privacy, integrity and security. These issues are addressed by the European Commission, in forms of recommendations and regulatory requirements. The case described in this paper as a proof-of-concept for our reference architecture format is a result of EU's focus on energy, and especially advanced meter systems and their security. The case is a part of larger EU project where one of the goals was to produce an AMI reference architecture for cyber security analysis.

Basically, there are no existing reference models for security analysis and AMI. However, there are some reference models for AMI systems and these have been used as a foundation for the reference model created in the project, see Fig. 5.

The reference model was implemented using securiCAD [12], as mentioned in chapter IV-B.

A. The AMI reference model

Primarily, the creation of the AMI security reference

model was done by studying various available reference architectures for AMI incl. reports from different AMI-system vendors and research on AMI, AMI-security, and, smart grids. One such source of inspiration can be seen in Fig. 5. The modeling was a four/five step iterative process; 1) logical level incl. systems and dataflows, 2) protocols and software products, 3) users, user accounts, and access control, 4) network layer incl. routers, firewalls, IDS, and IPS, and finally adding the 5) attacker(s) for the simulation and scenarios.

The AMI reference model is divided into three levels of abstraction: i) Network level with entities like; networks zones, routers, firewalls, zone management, IDS, and IPS. ii) Logical level with all the IT-systems, OT-systems, (incl. operating system, client, and server), data store, user, user account, and access control. iii) Data flows with associated protocols.

In order to grasp the full model one would need to install securiCAD and browse the different views. In this paper we have chosen one view in order to exemplify what a reference model and its accompanying analysis could look like and how it could be used. In Fig. 6 we present the Meter Management System (MMS) view. The MMS manages meter data and meter operations to enable advanced metering infrastructure (AMI). A MMS usually has a great flexibility for operating system platforms, database selection, data acquisition approaches, look and feel, reporting, and alarm capabilities. This view also contains the AMI Forecasting System (AMI FS), which has the role to collect and process forecast data to the AMI. There is also a connection between MMS and the EAM being a lifecycle management system of physical assets of an organization. Included is also the Customer Information System (CIS) collecting customer notifications, billing and payment information/confirmations, and service order request from customers, as well as the Outage Management System (OMS) handling outage record requests, activity records, and planned outage info.

Of course you can use this kind of reference model to design your AMI system focusing on the functions and structure, and this is also what most reference models are for. However, the true value of a reference model comes when you can use it for analysis and making sure that the architecture proposed is actually any good. Being able to model and analyze before implementing your system can help you save time since you can test your design decisions before you actually implement them. In the next subsection we will demonstrate how the AMI reference model can be used for cyber security analysis.

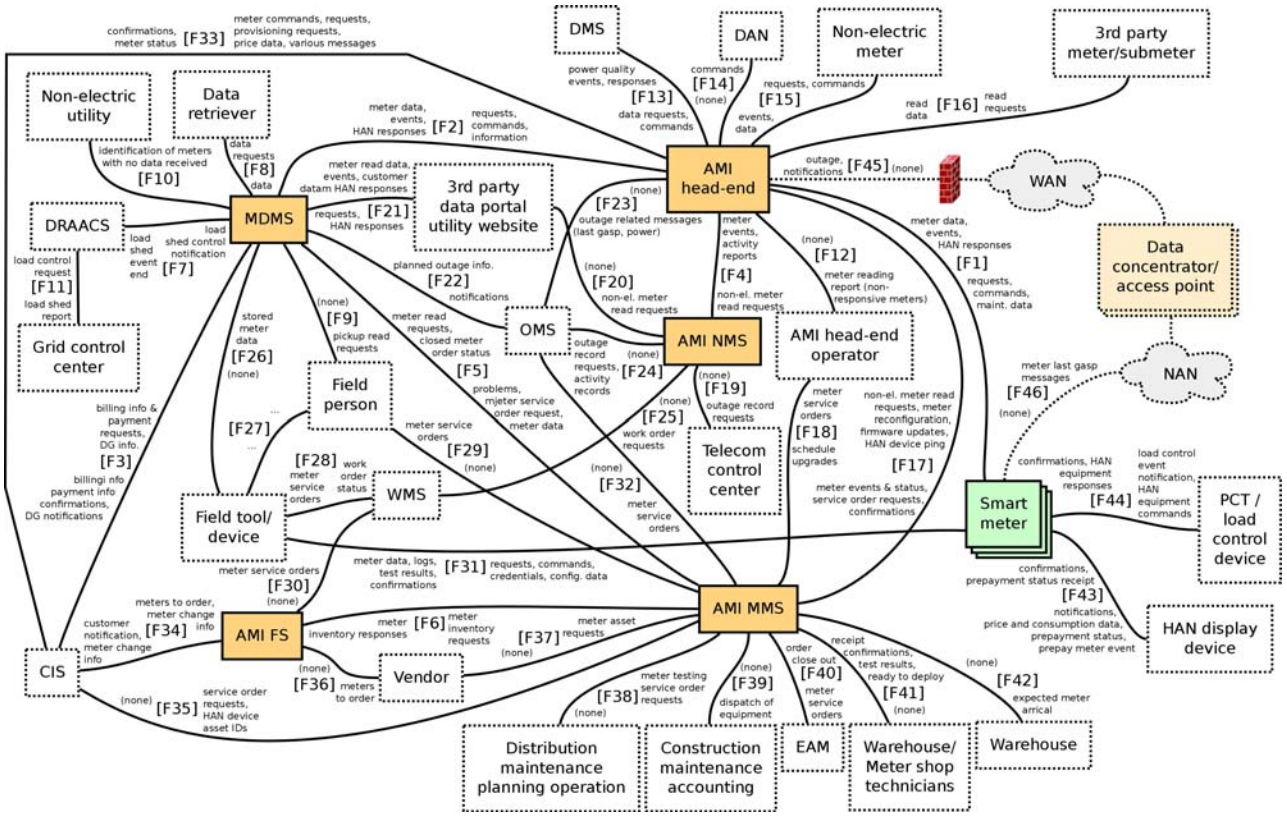

Fig. 5. One of two reference models used as a basis for the one proposed in our case.

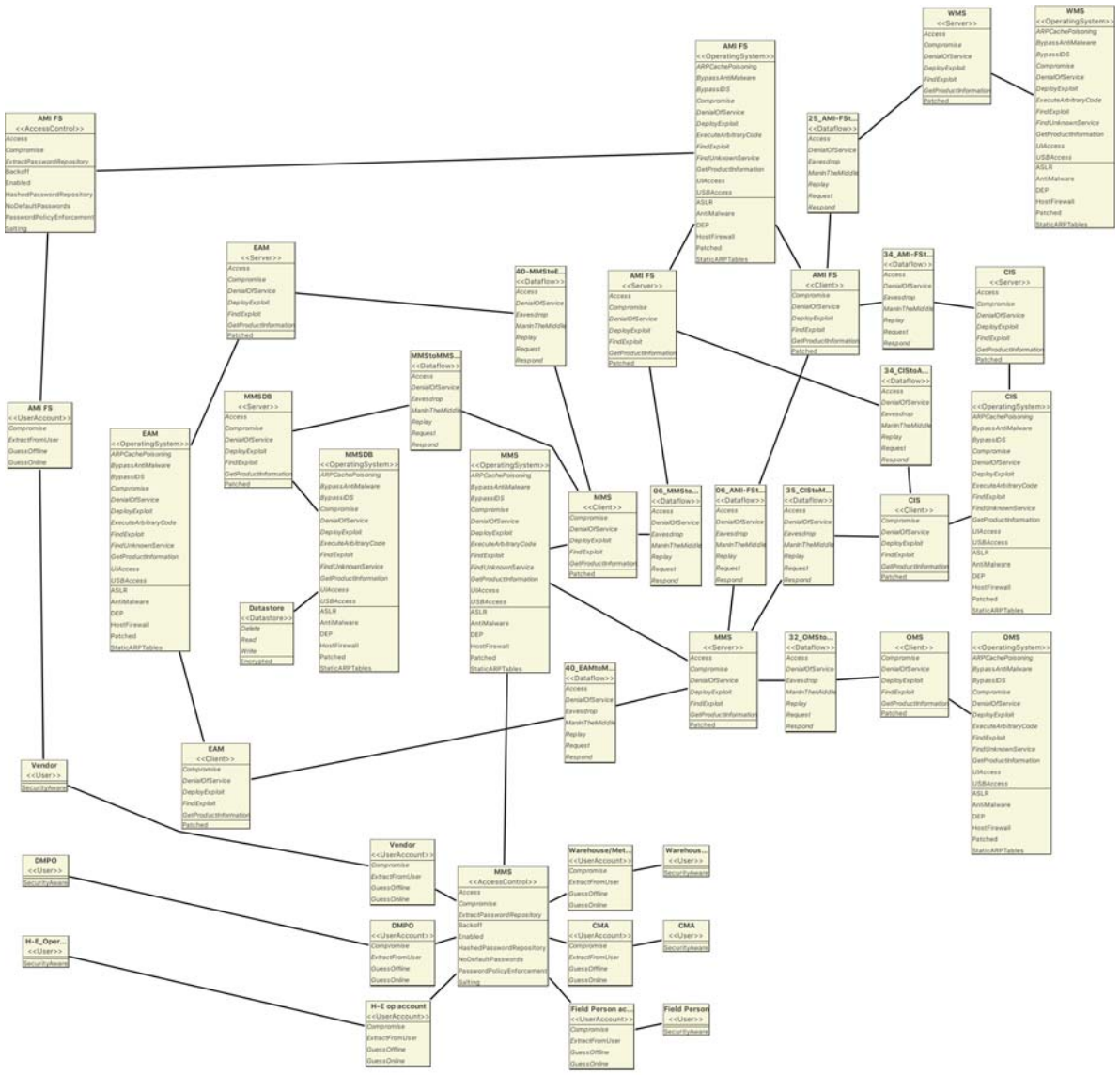

Fig. 6. An example view of the AMI reference model implemented in securiCAD.

B. Security analysis of an AMI system

In total we analyzed three different scenarios within the project; 1) attack from the public Internet, 2) attack from a customer, 3) structural changes to the architecture.

The results of these scenarios are reported with models, graphs, attack paths, and tables. The graphs show the time it takes to compromise (TTC) each part of the model, the time it takes for an attacker (professional penetration tester) to succeed with an attack on a chosen entity. Basically the graph illustrates the probability to succeed in X days, where the graph explicitly highlights how many days it is likely to succeed within 5%, 50%, and 95%.

In securiCAD, attack paths are presented in three different views; all paths, critical path, and fastest path. To limit the section somewhat, we have chosen to present the fastest path for some selected attributes. This is a good addition to the distribution graphs in order to understand the complexity of an attack.

The tables are used as a supplement for easier comparison of results. The tables should be interpreted the same way as the distribution graphs, described above.

The first scenario is an attack coming from the public Internet through an external server. The attacker's target in this scenario is the MDMS (Meter Data Management System) where the attacker wants to find sensitive customer data and the MMS (Meter Management System) where she can control a customer's meter in the sense it could be able to shut out customers from the grid and tinker with prepaid accounts.

With the heat map (Fig. 7 we can se that the attacker manages to penetrate more or less the whole system (all objects in the model have attributes that are marked as red meaning it is possible for an attacker to succeed with that

particular attack within a certain threshold).

Table I shows some of the TTC numbers, e.g. a man in the middle attack of the dataflow between the MMS and the MMS database succeeds in 5% of the attempts given 4 days, while given 37 days 95%. The same information can also be shown by the distribution graphs, as in Fig. 8.

In the AMI case we have one large reference model describing the advanced metering infrastructure according to commonly well-known functional reference models available. Doing a cyber security analysis using different possible attack scenarios shows what weaknesses one would have if implementing the reference architecture as it is with out any modifications. Since this is a reference architecture most companies would do some changes to it before implementation, both functionally and non-functionally. Looking at scenario 1 in our analysis, we can clearly see that changes need to be made in regards of security. The next step would thus be to make appropriate changes to the architecture and run the analysis again to see if there are improvements and how much of an improvement it is. One could also test different changes to decide which one(s) is more economically efficient to implement.

Running our three scenarios and exploring these helped us compiling a list of recommendations, an excerpt is found below:

- Divide the networks into smaller zones, making it much more difficult for an attacker to penetrate the system when she has to pass more routers with corresponding firewalls IDSs, IPSs, and networks management policies.

TABLE I. SELECTED VALUES FROM THE TTC CALCULATIONS OF SCENARIO 1.

Type	**Object**	**Attack**	**5%**	**50%**	**95%**
Dataflow	MMStoMMSDB	ManInTheMiddle	4	11	37
OperatingSystem	MMS	Compromise	4	11	37
OperatingSystem	MMSDB	Compromise	4	11	37
OperatingSystem	MDMS	Compromise	4	11	37
OperatingSystem	H-E	Compromise	10	22	100[a]

[a] The upper limit was set to 100, meaning we are not interested in attacks that take longer than this.

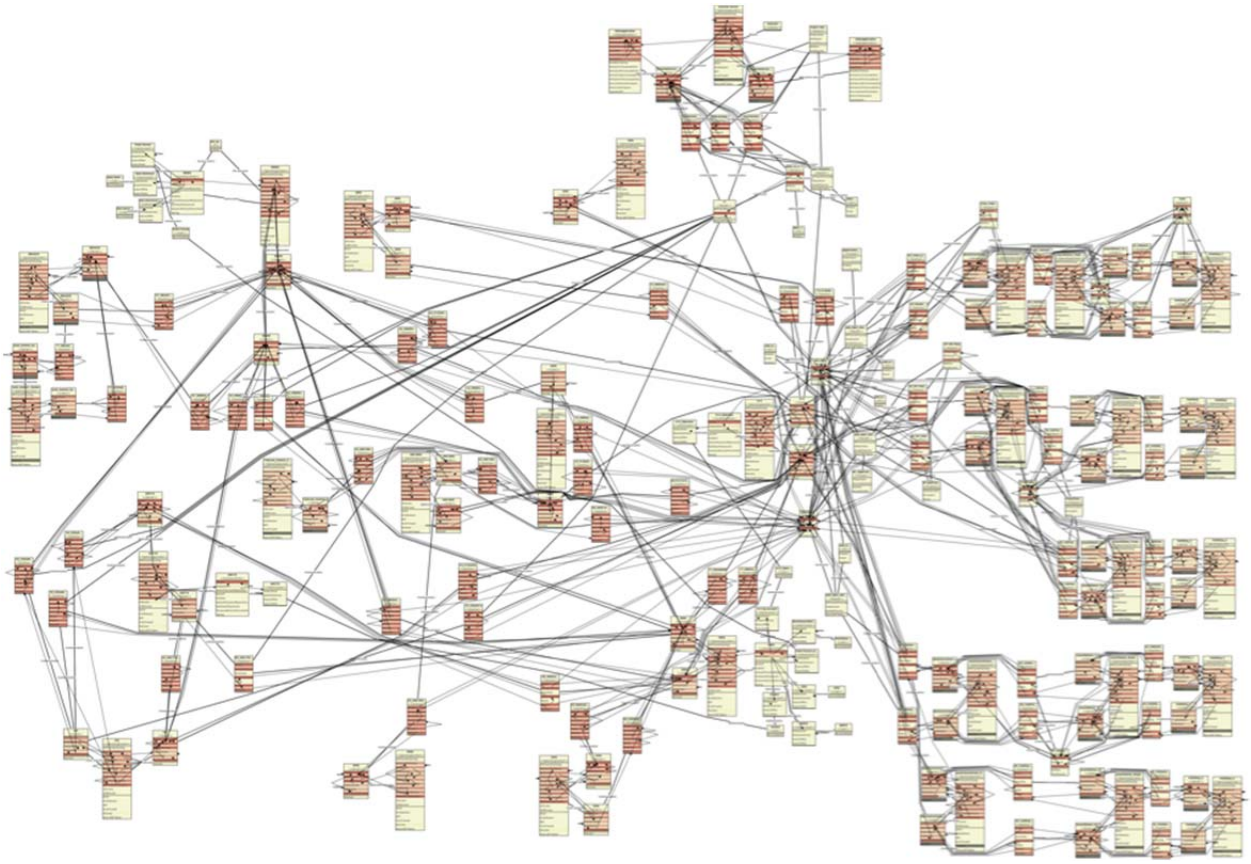

Fig. 7. A model illustrating the attack on an AMI system from the public Internet.

- Only connect systems to the larger networks if they really need to be there.
- Fast and reliable patch management, according to our analyses the attacker often uses vulnerabilities in patches in order to compromise systems.
- Use high security policies in access control linked to the most sensitive systems, by using defenses like back off, hash password repository, no default passwords, password policy enforcement, and salting many attacks can be delayed (to the extent of non-success).

Although the list of recommendations above seem simple and standardized, this is not case. We see endless of examples in industry where these simple security measures are neglected. Not only can the model and the analyses show this in a quantitative way, the approach can also help prioritize which or in what order to implement these and other.

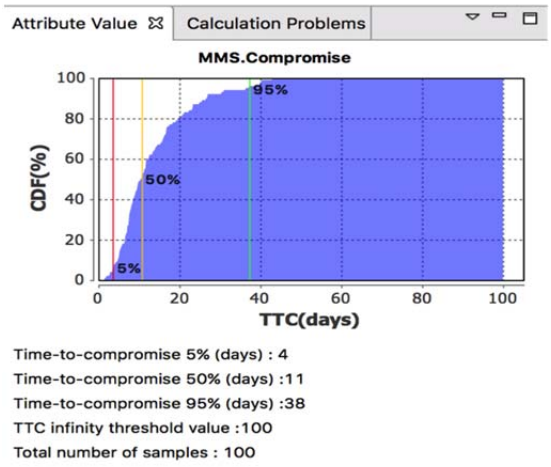

Fig. 8. The graph output from securiCAD illustrating the TTC for compromising the MMS.

To summarize the AMI case, we saw in this project (as in many other) that few have architecture models of their systems on a holistic level (like EA). When they do, these models do not contain details enough in order to analyze issues like security and thus they are often of very little use in the end. Mostly the models are used for communication between different business units as a way of aligning vocabulary. One reason often given by our industry partners is that it is too costly and difficult to collect the information needed for such a model. We believe this is one of the main reasons we see increased attention for techniques aiding data collection and modeling, including the use of reference models. As we discussed above the available reference models mainly focus on functional aspects, and structure of systems and infrastructure. The particular case described in this paper, AMI, shows us that the proposed reference model suggests an architecture that in no way is secure enough for todays requirements on smart meter systems. By running a series of analyses we could show this and provide recommendations. By using our reference model with an appropriate tool the companies can themselves run more analyses and decide what modifications to implement. The reference model, in the format proposed in this paper, helps them start much faster than if they would need to start from scratch and / or do manual analysis using expensive and difficult to find manual human expertise.

IX. DISCUSSION

The main design goals for the reference model format were high interoperability, expressiveness and ease of use. Thanks to the ability to represent uncertainty, the format's expressiveness exceeds that of the sole combination of UML and OCL, although considerably lower than that of free text with free drawings. A degree of interoperability stems from full backward compatibility with the combination of UML and OCL.

Finally, ease of use is supported by the format's simplicity, interoperability and flexibility through the script-like instantiation.

In spite of the technical scope of the case shown in section VIII, it is possible to apply the format of reference models to arbitrary levels of EA models including business architecture, customer context [10] and even type 2 architectures [5] (life cycle architectures), given their admissibility to modeling using UML syntax.

Relating back to the categories of reuse discussed in chapter III, the proposed format of reference models aims to primarily contribute through the following: (1) decreased cost and increased benefits of reusing EA models – both within and across organizations; (2) enabling modelers, especially those who produce reference models, to alleviate the consumers' abstraction challenge [44] by abstracting from very concrete snapshots of architectures through including uncertainty in values and structure; or including the degree of abstraction as a parameter of the EA analytical reference model, as long as the technical equipment (i.e., EA modeling and analysis toolset) allows doing so; (3) supporting specialization [43] by the possibility to arbitrarily modify the instantiations of reference models; and (4) integration (ibid.), although only indirectly, through the choice of highly adopted means of modeling (models corresponding to the syntax of UML class diagrams [38]) and extensions that are backwards compatible (P2AMF [24]). Important aspects of reuse that remain largely unaddressed by the proposed format is the composability of models [44], since an EA-analytical reference model using the format can still correspond to an arbitrary EA-analytical metamodel, which co-determines the composability. Another composability concern would be the case of two reference models corresponding to the same EA-analytical metamodel, but each using a different reference model format. Further, although the validity and credibility of reference models are properties that remain to be determined

and shown within the life cycles of each individual reference model, the format could both constrain achieving these properties directly through its limited expressiveness or feature set, or support the properties indirectly, to the extent it would lead to an increased reuse of EA models in an organization, community, or in general. Finally, the properties of abstraction and selection as described in [43] are seen as entirely external to the format of reference models, and hence remain not addressed by it.

The aim of using the rather large AMI reference model as a case was to demonstrate the potential of such reference models. This potential is not negligible for the practice of modeling and EA analysis, especially for evaluating to-be scenarios. On the other hand, it must be admitted that the degree of optimization could be lower for smaller and seldom used reference models; as well as for reference models requiring many additional modifications after instantiation, compared to the amount of effort it would save. In the future, more comprehensive and rigorous study of existing reference models, their formats and their use, would be beneficial.

Reference models typically serve to provide information to their user. However, as a documentation artifact, they also need to be updated, perhaps even frequently. To that end, two concepts with supportive potential are on the horizon – living models [7], and automated collection of EA data [8], [49]. Both might contribute to the ease of creating reference models and keeping them up-to-date. Other prospective future work on reference models include formulating concrete reference models suited for automated EA analysis, especially in the domains of smart grid and cyber security. Proposal of reference models from different domains, both industrial and office/corporate, would also be beneficial. Lastly, reference models for additional EA viewpoints than cyber security could be proposed (e.g., interoperability and availability), to the end of formulating multi-aspect reference models, combining data for multiple viewpoints and metamodels. Finally, the possibilities offered by multi-level modeling using clabjects [3] might be embraced to refine the supportive value of reference models.

X. CONCLUSIONS

In this paper we propose a format for reference models capable of expressing both exact quantities and structure, and stochastic uncertainty, and so able to support reuse in modeling for EA analysis. Using a simple case, we demonstrated that reference models using the format have the potential to save modeling effort. Seeing automated EA analysis as an emerging trend and an important cornerstone of efficient technology management, we believe that reference models will enrich EA practice and thus technology management, and that the proposed format will inspire steps toward higher interoperability between EA analytic approaches and solutions.

ACKNOWLEDGMENTS

This project has received funding from the European Unions Seventh Framework Programme for research, technological development and demonstration under grant agreement no. 607109 (SEGRID). Also, part of it has been financed by KIC InnoEnergy, SweGrids (www.swegrids.se), ÅForsk, and the SALVAGE project (Cyber-physical security for low-voltage grids) funded via ERA-Net SmartGrids programme.

REFERENCES

[1] Angelov, S., P. Grefen and D. Greefhorst, "A classification of software reference architectures: Analyzing their success and effectiveness," in *Software Architecture, 2009 & European Conference on Software Architecture. WICSA/ECSA 2009. Joint Working IEEE/IFIP Conference on*. IEEE, pp. 141–150, 2009.

[2] Atkinson, C. and T. Kuhne, "Model-driven development: a metamodeling foundation," *Software, IEEE*, vol. 20, no. 5, pp. 36–41, 2003.

[3] Atkinson, C., B. Kennel and B. Goß, "The level-agnostic modeling language," in *Software Language Engineering*. Springer, pp. 266–275, 2011.

[4] Atzori, L., A. Iera and G. Morabito, "The internet of things: A survey," *Computer Networks*, vol. 54, no. 15, pp. 2787-2805, 2010.

[5] Bernus, P., L. Nemes and G. Schmidt, *Handbook on enterprise architecture*. Springer, 2003.

[6] Bernus, P. and O. Noran, "A metamodel for enterprise architecture," in *Enterprise Architecture, Integration and Interoperability*. Springer, pp. 56–65, 2010.

[7] Breu, R., B. Agreiter, M. Farwick, M. Felderer, M. Hafner and F. Innerhofer-Oberperfler, "Living models-ten principles for change-driven software engineering." *Int. J. Software and Informatics*, vol. 5, no. 1-2, pp. 267–290, 2011.

[8] Buschle, M., M. Ekstedt, S. Grunow, M. Hauder, F. Matthes and S. Roth, "Automating enterprise architecture documentation using an enterprise service bus," in *AMCIS 2012 Proceedings,* paper 13, 2012.

[9] Cason Jr, W. C., C. F. Dalton, J. S. Morio, S. W. Reynolds, R. Renteria and S. Lemay, "Method and system for a reference model for an enterprise architecture," US Patent 7,890,545, Feb. 15 2011.

[10] Cloutier, R., G. Muller, D. Verma, R. Nilchiani, E. Hole and M. Bone, "The concept of reference architectures," *Systems Engineering*, vol. 13, no. 1, pp. 14–27, 2010.

[11] Ekstedt, M., P. Johnson and R. Lagerström, "Enterprise Architecture Modeling and Analysis of Quality Attributes–The Multi-Attribute Prediction Language (MAPL)," in *1st Scandinavian Workshop on the Engineering of Systems-of-Systems (SWESoS 2015)*, p. 10, 2015.

[12] Ekstedt, M., P. Johnson, R. Lagerström, D. Gorton, J. Nydren and K. Shahzad, "securiCAD by Foreseeti: A CAD Tool for Enterprise Cyber Security Management," in *Enterprise Distributed Object Computing Workshop (EDOCW), 2015 IEEE 19th International*. IEEE, pp. 152–155, 2015.

[13] Frakes, W. B. and K. Kang, "Software reuse research: Status and future," *IEEE transactions on Software Engineering*, no. 7, pp. 529–536, 2005.

[14] Franke, U., P. Johnson and J. König, "An architecture framework for enterprise it service availability analysis," *Software & Systems Modeling*, pp. 1–29, 2013.

[15] Galster, M. and P. Avgeriou, "Empirically-grounded reference architectures: a proposal," in *Proceedings of the joint ACM SIGSOFT conference–QoSA and ACM SIGSOFT symposium–ISARCS on Quality of software architectures–QoSA and architecting critical systems–ISARCS*. ACM, pp. 153–158, 2011.

[16] Gonzalez-Perez, C. and B. Henderson-Sellers, "A powertype-based metamodelling framework," *Software & Systems Modeling*, vol. 5, no. 1, pp. 72–90, 2006.

[17] Hevner, A. R., S. T. March, J. Park and S. Ram, "Design science in information systems research," *MIS Quarterly*, vol. 28, no. 1, pp. pp. 75–105, 2004.

[18] Holm, H., M. Buschle, R. Lagerstro¨m and M. Ekstedt, "Automatic data collection for enterprise architecture models," *Software & Systems Modeling*, vol. 13, no. 2, pp. 825–841, 2014.

[19] Holm, H., K. Shahzad, M. Buschle and M. Ekstedt, "P2CySeMoL: Predictive, probabilistic cyber security modeling language," *Computer and Information Science*, 2014.

[20] International Data Corporation, "Enterprise architecture: Strategic architecture for the 3rd platform," Retrieved 1/10/2016 World Wide Web, http://www.idc.com/getdoc.jsp?containerId=251163

[21] Irlbeck, M., D. Bytschkow, G. Hackenberg and V. Koutsoumpas, "Towards a bottom-up development of reference architectures for smart energy systems," in *Software Engineering Challenges for the Smart Grid (SE4SG), 2013 2nd International Workshop on*. IEEE, pp. 9–16, 2013.

[22] Johnson, P., R. Lagerström, P. Närman and M. Simonsson, "Enterprise architecture analysis with extended influence diagrams," *Information Systems Frontiers*, vol. 9, no. 2-3, pp. 163–180, 2007.

[23] Johnson, P., R. Lagerström, M. Ekstedt and M. Ö sterlind, *IT Management with Enterprise Architecture*. Stockholm, Sweden: KTH Royal Institute of Technology, 2012.

[24] Johnson, P., J. Ullberg, M. Buschle, U. Franke and K. Shahzad, "P2AMF: Predictive, probabilistic architecture modeling framework," in *Enterprise Interoperability*. Springer, pp. 104–117, 2013.

[25] Johnson, P., J. Ullberg, M. Buschle, U. Franke and K. Shahzad, "An architecture modeling framework for probabilistic prediction," *Information Systems and e-Business Management*, pp. 1–28, 2014.

[26] KTH, Industrial Information and Control Systems, "Multi-Attribute Prediction (MAP) class diagram," Retrieved 1/10/2016 World Wide Web, https://www.kth.se/en/ees/omskolan/organisation/avdelningar/ics/research/sa/the-multi- attribute-prediction-map-class-diagram-1.387306

[27] Lagerström, R., U. Franke, P. Johnson and J. Ullberg, "A method for creating enterprise architecture metamodels–applied to systems modifiability analysis," *International Journal of Computer Science and Applications*, vol. 6, no. 5, pp. 89–120, 2009.

[28] Lagerström, R., P. Johnson and D. Höök, "Architecture analysis of enterprise systems modifiability–models, analysis, and validation," *Journal of Systems and Software*, vol. 83, no. 8, pp. 1387–1403, 2010.

[29] Lagerström, R., T. Sommestad, M. Buschle and M. Ekstedt, "Enterprise architecture management's impact on information technology success," in *System Sciences (HICSS), 2011 44th Hawaii International Conference on*. IEEE, pp. 1–10, 2011.

[30] Lagerström, R., C. Baldwin, A. MacCormack and D. Dreyfus, "Visualizing and measuring enterprise architecture: An exploratory biopharma case." Springer, 2013.

[31] Lankhorst, M.; *Enterprise architecture at work: Modelling, communication and analysis*. Springer, 2013.

[32] Liu, F., J. Tong, J. Mao, R. Bohn, J. Messina, L. Badger and D. Leaf, "NIST cloud computing reference architecture," *NIST special publication*, vol. 500, p. 292, 2011.

[33] MacCormack, A., R. Lagerström and C. Y. Baldwin, "A methodology for operationalizing enterprise architecture and evaluating enterprise IT flexibility," *Harvard Business School Finance Working Paper*, pp. 15–60, 2015.

[34] Martinez-Fernandez, S., C. Ayala, X. Franch and H. M. Marques, "Artifacts of software reference architectures: A case study." EASE, 2014.

[35] Milch, B., B. Marthi, S. Russell, D. Sontag, D. L. Ong and A. Kolobov, "1 blog: Probabilistic models with unknown objects," *Statistical relational learning*, p. 373, 2007.

[36] Nakagawa, E. Y., F. Oquendo and M. Becker, "Ramodel: A reference model for reference architectures," in *Software Architecture (WICSA) and European Conference on Software Architecture (ECSA), 2012 Joint Working IEEE/IFIP Conference on*. IEEE, pp. 297–301, 2012.

[37] Närman, P., P. Johnson, R. Lagerström, U. Franke and M. Ekstedt, "Data collection prioritization for system quality analysis," *Electronic Notes in Theoretical Computer Science*, vol. 233, pp. 29–42, 2009.

[38] Object Management Group, "Unified modeling language (UML)," Retrieved 1/10/2016 World Wide Web, http://www.omg.org/spec/UML/2.4.1/

[39] Object Management Group, "Object Constraint Language (OCL)," Retrieved 1/10/2016 World Wide Web, http://www.omg.org/spec/OCL/2.4/

[40] Object Management Group. "Meta Object Facility (MOF)," Retrieved 1/10/2016 World Wide Web, http://www.omg.org/spec/MOF/2.4.2/

[41] Open Group, "ArchiMate 2.0 Specification, Technical Standard," Reading, UK: The Open Group, Retrieved 1/10/2016 World Wide Web, http://www.opengroup.org/archimate/

[42] Open Networking Foundation, "Software-defined networking: The new norm for networks," Retrieved 1/10/2016 World Wide Web, https://www.opennetworking.org/images/stories/downloads/sdn-resources/white-papers/wp-sdn-newnorm.pdf

[43] Pidd, M., "Reusing simulation components: simulation software and model reuse: a polemic," in *Proceedings of the 34th conference on Winter simulation: exploring new frontiers*. Winter Simulation Conference, pp. 772–775, 2002.

[44] Robinson, S., R. E. Nance, R. J. Paul, M. Pidd and S. J. Taylor, "Simulation model reuse: definitions, benefits and obstacles," *Simulation modelling practice and theory*, vol. 12, no. 7, pp. 479–494, 2004.

[45] Sheyner, O., J. Haines, S. Jha, R. Lippmann and J. M. Wing, "Automated generation and analysis of attack graphs," in *Security and privacy, 2002. Proceedings. 2002 IEEE Symposium on*. IEEE, pp. 273–284, 2002.

[46] Sommestad, T., M. Ekstedt and H. Holm, "The cyber security modeling language: A tool for assessing the vulnerability of enterprise system architectures," *Systems Journal, IEEE*, vol. 7, no. 3, pp. 363–373, Sept 2013.

[47] Sprinkle, J., B. Rumpe, H. Vangheluwe and G. Karsai, "Metamodelling: State of the art and research challenges," in *Model-Based Engineering of Embedded Real-Time Systems*. Springer, pp. 57–76, 2011.

[48] Ullberg, J., P. Johnson and M. Buschle, "A language for interoperability modeling and prediction," *Computers in Industry*, vol. 63, no. 8, pp. 766–774, 2012.

[49] Välja, M., R. Lagerström, M. Ekstedt and M. Korman, "A requirements based approach for automating enterprise it architecture modeling using multiple data sources," in *Enterprise Distributed Object Computing Workshop (EDOCW), 2015 IEEE 19th International*. IEEE, pp. 79–87, 2015.