Improving IT Infrastructures Representation: A UML Profile

Luís Ferreira da Silva[1]
luis.alexandre@campus.fct.unl.pt

Fernando Brito e Abreu[2,1]
fba@iscte-iul.pt

Victor Moreira[2]
vitor_hugo_moreira@iscte.pt

[1]QUASAR Group, CITI, FCT/UNL
Universidade Nova de Lisboa
2829-516 Caparica, Portugal

[2]DCTI, ISTA, ISCTE-IUL
Instituto Universitário de Lisboa
1649-026 Lisboa, Portugal

***Abstract*— IT infrastructures are most times informally modeled. The resulting models are ambiguous to stakeholders, cannot be checked for validity, and therefore are unable to play their important role in design, deployment and maintenance activities. The main reason for such a poor state-of-the-art lies mainly in the absence of a modeling language capable of representing IT infrastructures at the required level of abstraction. Indeed, existing candidate languages are too abstract, as shown in this paper by reviewing their metamodels. The present paper mitigates this problem by proposing a UML profile to describe the semantics of an IT infrastructure.**

Keywords – Information Technology; IT Infrastructures; UML Profile; Modeling; Design Patterns

I. INTRODUCTION

An *Information Technology Infrastructure (ITI)*, also known as *Technology Architecture* in most Enterprise Architecture frameworks, is the foundation on which business processes that drive the success of an organization are based [1] and has been defined as "*all hardware, software, networks and facilities, etc. that are needed to develop, test, deliver, monitor, control or support IT services*" [2]. Some of the unique characteristics of the ITI layer are:

- Is the foundation for all the other architecture layers, meaning that a problem at this layer can influence all the layers above;
- Provides services that are used by multiple applications, processes, and users;
- Is usually not perceived as a layer that offers financial benefits, but rather as an utility that enables business processes to be performed.

The ability to streamline the conception, deployment and maintenance of ITIs depends largely on our ability to model them, as in most engineering endeavors, to produce complex artifacts. The use of models was introduced in Computer Science in the seventies to simplify complexity in Software Engineering. Models convey a simplified representation of part of the real world, which can be useful for analytical purposes. The use of models provides a way to view specific aspects of a system with multiple levels of abstraction within different contexts.

To produce models, we require a modeling language providing a set of composable constructs. The use of informal modeling notations creates communicational problems among stakeholders and ultimately makes ITIs suffer the same problems of legacy software: undocumented decisions, redundancy, inconsistencies and increased cost of ownership. To mitigate these issues, we should adopt a well-formed graphical notation, based on a formal grammar, usually called a *metamodel*. The latter includes precise definitions of constructs and their relationships, along with composition rules that must be fulfilled for creating valid models. Models are said to be metamodel instances since they conform to it. Metamodels allow the development of syntax checking editors and validation tools. With such a support, models are then prone to provide a less ambiguous and shared meaning to all relevant stakeholders, and therefore play their expected role in ITI engineering.

Despite the aforementioned benefits on using a precise modeling language, our experience in the field has shown that most organizations depict their ITIs informally, either using some ad-hoc templates or informal notation not supported by a standard or framework, resulting in ambiguous models without any kind of traceability features like the one represented in Figure 1. Such ad-hoc models frequently lead to discussions as each stakeholder has its own interpretation.

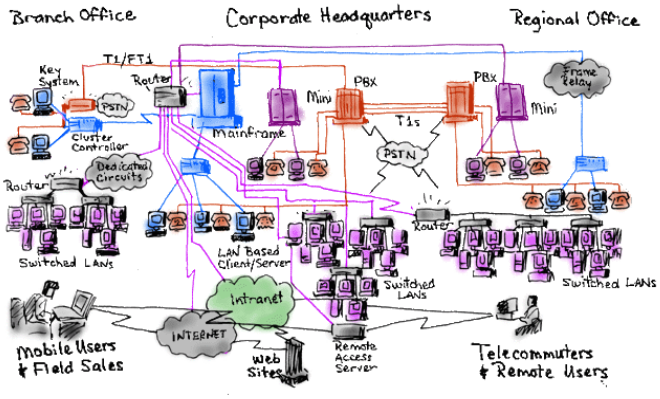

Figure 1. Model of an IT infrastructure (source: [3])

In this paper, we propose an extension to the UML2 metamodel, provided as a profile, to describe the semantics of ITI modeling constructs.

 ISBN: 978-1-61208-230-1

This paper is organized as follows: in the next Section we review related work; in Section III we present the UML profile itself, by overviewing its structure and stereotypes; then, in Section IV, we briefly describe how we have deployed the proposed profile in a professional modeling tool; finally, in Section V, we present some conclusions and future work.

This paper extends (by presenting related work and a modeling example using the profile) and improves (by providing more detail on the profile and on the future work) a previous short paper of ours [4].

II. RELATED WORK

Several modeling languages provide constructs for modeling ITIs with some precision. In this section, we will overview those constructs, as defined in the metamodels of three of the most well-known languages that can be used in the context of IT infrastructures: *UML*, *TOGAF* and *ArchiMate*.

A. UML Metamodel

UML is a general-purpose modeling language embodying a collection of best engineering practices that have proven successful in the modeling of large and complex systems of a wide range of domains. Under the stewardship of the Object Management Group (OMG), UML has emerged as the software industry's dominant modeling language. IT infrastructures are modeled in UML with *Deployment Diagrams*. The latter allows representing the hardware for a system, the software that is installed on that hardware, and the middleware used to connect machines.

Since UML version 2 (UML2) has thirteen different types of diagrams, our first endeavor was assessing their relative usage at a global scale, based upon the hits provided by several web search engines, either general purpose, or academic / research oriented. Plotted values in Figure 2 are represented in percentage of total hits. When available, we split textual search hits from image search hits, but the ranking in both cases does not differ significantly.

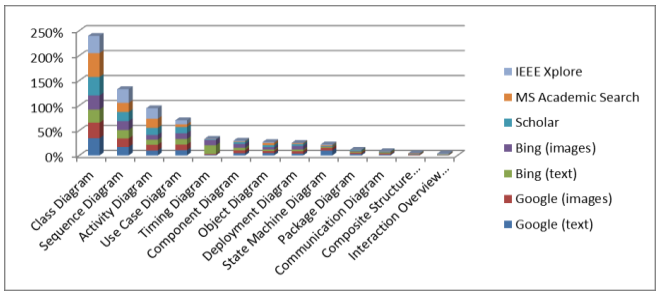

Figure 2. Ranking the use of the thirteen UML2 diagrams

As it can be observed in Figure 2, *Deployment Diagrams* are among the less used UML2 diagrams. A possible interpretation for this phenomenon is that UML2 offers limited modeling constructs (e.g., nodes, components and associations), that do not cope "as is" with the required diversity for modeling ITIs.

There are four modeling elements in UML deployment diagrams, represented as metaclasses in the corresponding UML metamodel extract, as shown in Figure 3 and Figure 4. Those constructs are: *Nodes* to represent a hardware component, *Components* to represent software, *Dependencies* to show that one component relies upon another component and *Links* to connect nodes.

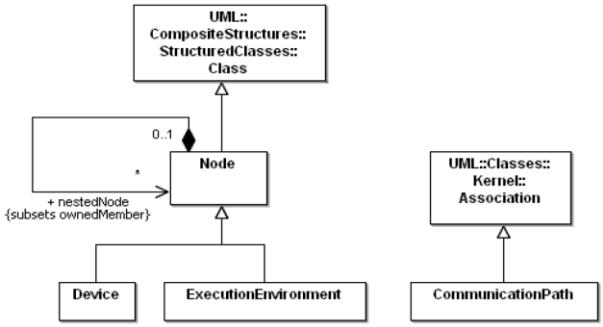

Figure 3. UML metamodel extract corresponding to deployment diagrams.

As can be seen in Figure 3, *Node* is a central modeling element. It can have other elements of type *Node* and represents the environment in which a component or a set of components execute. A *Node* is a generic concept and can represent several things such as a physical hardware device, an operating system or infrastructure software (e.g., database server, web server, application server) and is connected through communication paths.

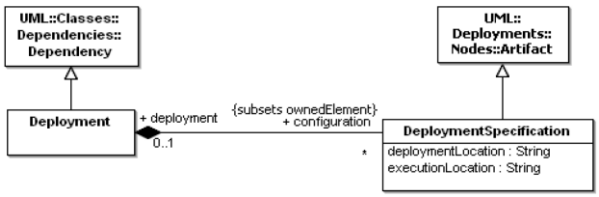

Figure 4. Metaclasses used to define the deployment component.

UML2 has a comprehensive coverage of the whole lifecycle in software development. As a result, its large specification, spanning more than 900 pages [5, 6] is in some aspects too abstract. This is well the case of *Deployment Diagrams*, and as a result they are not widely used as other UML2 diagrams, as corroborated by our survey. In short, "plain vanilla" UML provides no specialized stereotypes for the many concepts and association types used in any IT infrastructure, what makes it a weak candidate for modeling ITIs.

B. TOGAF Content Metamodel

The *Open Group Architecture Framework* (TOGAF) is a framework for enterprise architecture developed by the *Open Group Architecture Forum* in the United States,

which provides a comprehensive approach for designing, planning, implementation, and governance of an enterprise information architecture. TOGAF is a high level and holistic approach to design, which is typically modeled at four levels: Business, Application, Data, and Technology. It tries to give a well-tested overall starting model to information architects, which can then be built upon. It relies heavily on modularization, standardization and already existing, proven technologies and products. Its latest release, at the time of writing, also includes a metamodel called *Content Metamodel*, that defines all types of building blocks that exist within architecture and how they are related to each other to allow architectural concepts to be captured, stored, filtered and queried in a structured and consistent manner. The IT infrastructure (called technology architecture in TOGAF terminology) is part of the *Content Metamodel* and its direct relationships are shown in Figure 5.

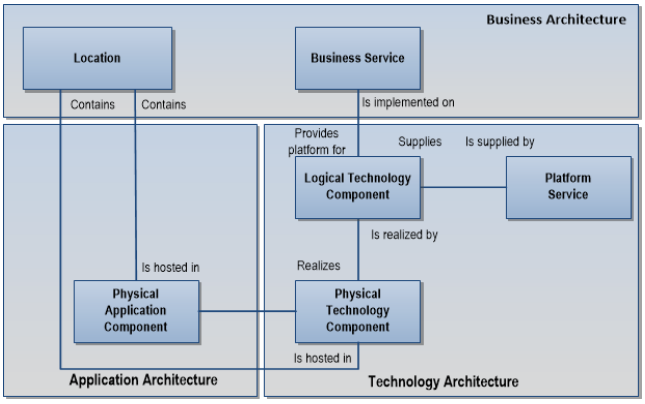

Figure 5. TOGAF 9.1 metamodel extract

From an IT infrastructure perspective, there are few concepts in the content metamodel: the *Platform Service* that represents the support for delivering applications, the *Logical Technology Component* used to represent a class of technology products and *Physical Technology Component* to represent specific technology products. As with UML, these general concepts from TOGAF's *Content Metamodel* allow, in theory, to model IT infrastructures. The lack of specialized stereotypes and relationships appears to be a serious hindrance for its effective adoption. Furthermore, some authors argue that TOGAF's *Content Metamodel* lacks a formal ontology to mitigate its ambiguities and inconsistencies [7].

C. *Archimate Metamodel*

ArchiMate is an open architecture modeling standard with focus on the visualization of viewpoints and notations on models. The metamodel encompasses several enterprise architecture domains (*Business, Application, Information, Technology*).

The *ArchiMate* metamodel was inspired in the UML 2.0 standard [5, 6]. As seen in Figure 6, *Node* is also the main structural concept and is specialized in *Device* (e.g., servers) and *System Software* (e.g., operating system called "execution environment" in UML).

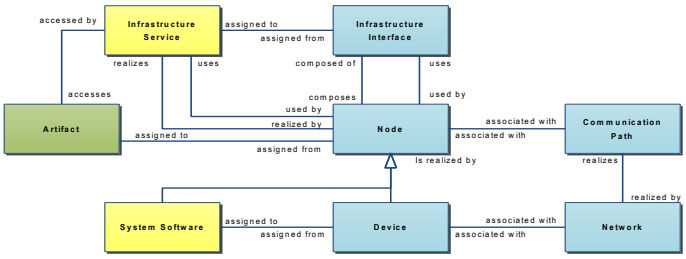

Figure 6. *ArchiMate* metamodel extract corresponding to IT Infrastructure modeling

The *Infrastructure Interface* is the "logical" location where the *Infrastructural Services* offered by a *Node* can be accessed by other *Nodes*. The *Communication Path* and *Network* are used to connect interrelated components in the technology layer. The *Artifact* (also taken from UML 2.0) represents a physical piece of information and can be deployed to a *Node*.

ArchiMate's technology architecture metamodel extract is more detailed than the corresponding TOGAF extract, namely by allowing to model the hardware platforms and communication infrastructure. However, it is still too generic and with more focus on describing the relationships between layers than providing clear guidelines and rules on how to model the various components of the technology architecture. It is argued in the *ArchiMate* [8]specification that modeling infrastructure components such as routers or database servers would add a level of detail that is not useful at the enterprise level of abstraction .

III. UML PROFILE FOR IT INFRASTRUCTURES

UML makes provisions for its own extension, by allowing "customization" to a specific area or domain, with a so-called *UML Profile*. The latter is a coherent collection of UML extensions (stereotypes, tagged values, and constraints) that allows refining the standard semantics in strictly additive manner (i.e. without contradicting it). For instance, a profile may use a stereotype to refine the concept of *Node*.

Several UML profiles have been proposed in the literature and some of them have been endorsed by the OMG itself. Examples include a profile for aspect-oriented software development [9], a profile for requirements management of software and embedded systems [10], a profile for business process modeling [11] and a profile for modeling real-time embedded systems [12].

According to Frank Ulrich, the existing tools and methods for IT management are not suitable because they focus on issues such as hardware and operational metrics. This author claims further that there is a gap between the technical level and IT management. He points out that a mitigating strategy to cope with the complexity of this task

 ISBN: 978-1-61208-230-1

would be providing adequate support for the analysis and communication among the various stakeholders. He corroborates our observation that existing approaches are too generic, therefore not providing the required level of detail and stresses that the lack of formality not only leads to communication problems among stakeholders, but also limits the use of automatic problem detection or validation techniques in existing infrastructures [13].

Due to the aforementioned limitations of existing modeling languages, we have developed an UML profile for IT infrastructures. The decision for extending UML, instead of developing a domain specific language (DSL) from scratch, was based on the following rationale:

- There is a large community, both in industry and academia, that understands and actually uses the UML language;
- Extending UML allows reusing existing UML modeling elements, with well-defined syntax and semantics;
- There are tools that support the development of UML profiles.

A. *ITI Profile Structure*

Figure 7 presents a conceptual view of the ITI profile, where the software and hardware layers are represented with different colors.

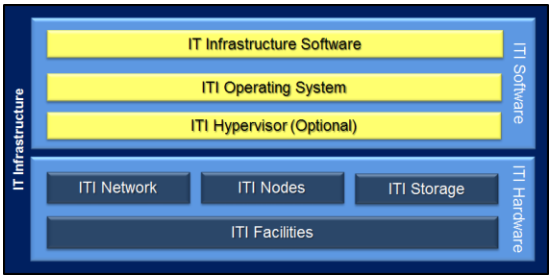

Figure 7. Layered structure of the ITI profile

The ITI Software Layer (Yellow) has three packages: *ITI Hypervisor*, *ITI Operating System* and *ITI Software,* while the ITI Hardware layer (Blue) has four packages: *ITI Facilities*, *ITI Network, ITI Nodes* and *ITI Storage.*

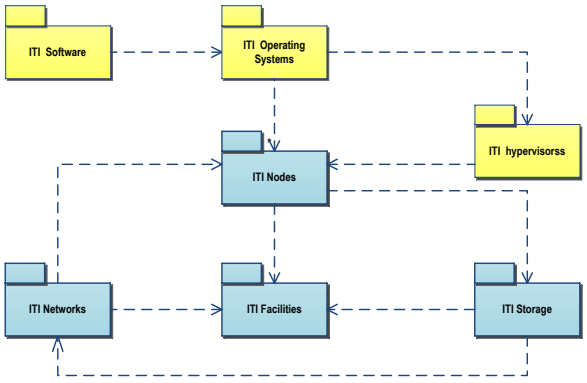

Figure 8. ITI packages and their relationships

Figure 8 provides an overview of the ITI packages and their relationships in the ITI profile. In the software layer the package *ITI Software* models software platforms such as antivirus, application servers, backup, collaboration servers, database servers, directory, and email servers, among others. These ITI platforms execute upon an *ITI Operating System* such as Windows, Linux, AIX. The operating system may be deployed directly on hardware or it can be deployed on top of an *ITI Hypervisor* such as XEN, Hyper-v or VmWare, that executes directly on the *ITI Nodes* hardware.

The package *ITI Nodes* is a very important one since it contains the constructs used to model systems such as servers and their components. The metaclass *Host* inherits the properties of an UML2 *Node* and was created with a set of stereotypes to allow the representation of physical and virtual servers, mainframes or supercomputers. The metaclass *Device* is similar to *Host* but is used to represent other equipment such as phones, tablets, slates, laptops, or PDAs. The *Peripheral* metaclass represents the components that may be connected to a *Host* or *Device* and includes monitors, keyboards, mice, printers or smartcard readers, among others. A *Port* is a built-in component in a *Host* or *Device* such as a host-based adapters or a network card.

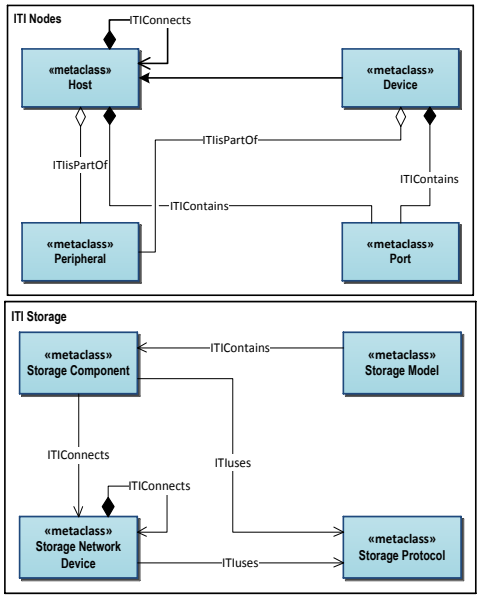

Figure 9. ITI Nodes and ITI Storage

The package *ITI Storage* represents the multiple *Storage Components* such as storage LUNs, storage arrays and pools, storage controllers and they may be configured in different *Storage Models* such as Storage Area Network (SAN) or Network Access Storage (NAS). These storage components are connected to *Hosts* and *Devices* trough *Storage Networks* using fiber channel or Ethernet routers or switches that use specific *Storage Protocols* such as ISCSI, Fiber Channel or Fiber Channel over Ethernet (FCoE) among other protocols. Both *ITI Nodes* and *ITI Storage* packages and their relationships are represented in Figure 9.

 ISBN: 978-1-61208-230-1

Hosts and *Devices* from the *ITI Nodes* package can be interconnected by *Network Devices* available in the *ITI Network Package*. *Network Devices* includes, among others, devices such as access points, firewalls, hubs, routers and switches. Those devices are used to create different *Network Zones* such perimeter networks, intranets or extranets and they communicate using a specific *Network Protocol* such as frame relay or Ethernet. The *Network Devices* may be configured in multiple *Network Types* such as LANs, WANs or Wi-Fi. All aforementioned components reside in the *ITI Facilities* package. Both packages are shown in Figure 10.

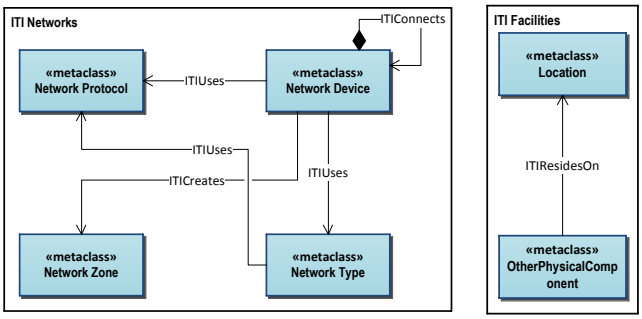

Figure 10. ITI Network and ITI Facilities

B. *ITI Profile Metaclasses and Stereotypes*

To connect multiple ITI components, we require ITI connectors that extend the metaclass *Association* or the metaclass *Composition*, as represented in Figure 11.

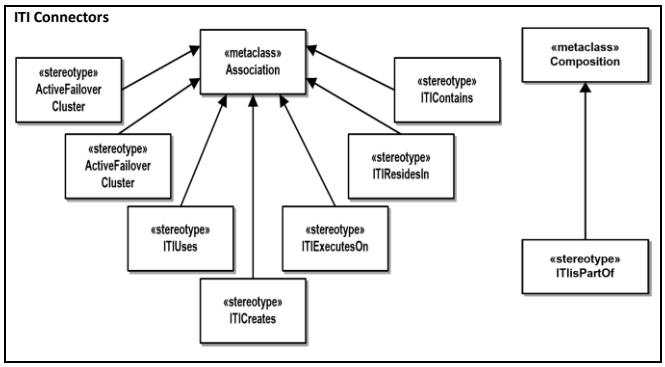

Figure 11. IT Infrastructure Connectors Package.

Besides these two connector metaclasses, we also extended UML2 metaclasses elements such as *Class, Location, Boundary, Device* and *Node*. An example is the package *ITI Facilities,* composed by the metaclass *Location* and the metaclass *OtherPhysicalComponents*. A location can be the *Headquarters*, a *Datacenter, a Branch Office,* or a *Regional Office. OtherPhysicalComponents* includes *Cables* to connect hosts, *Racks* to attach servers, *Power* supplies and *Cooling* systems. Both metaclasses and their extending stereotypes can be seen in Figure 12.

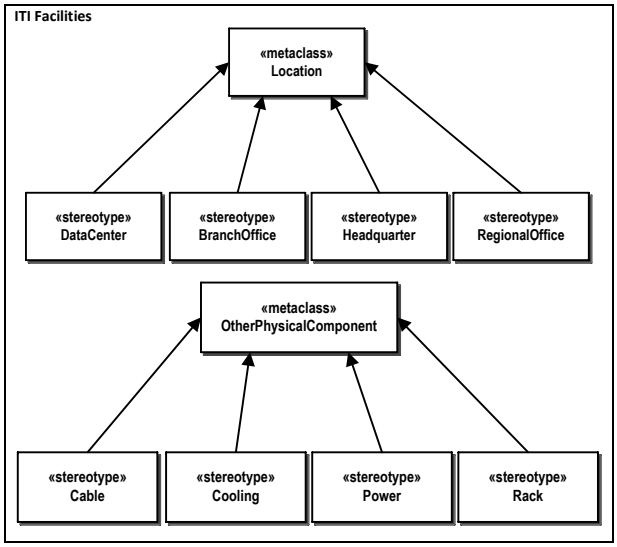

Figure 12. Package Facilities

To allow expressing as much information as desired in the ITI domain, we enriched each stereotype with additional attributes (called "tagged values" in earlier UML versions). The attributes chosen for each stereotype were based on our field experience and inspired on the standard *Common Information Model* (CIM) [14] created by the *Distributed Management Task Force* (DMTF). The latter is a worldwide initiative spearheaded by industry-leading technology companies such as AMD, Broadcom Corporation, CA, Cisco, Citrix Systems, EMC, Fujitsu, HP, Huawei, IBM, Intel, Microsoft, NetApp, Oracle, RedHat, SunGard and VMware.

CIM was created to provide a common approach to the management of systems, networks, applications and services and enable multiple vendors to exchange semantically rich management information between systems throughout the network. This paper only includes a subset of the stereotypes. The complete set of stereotypes, tagged values and constraints will be available as a technical report on the QUASAR group website [15].

IV. DEPLOYING THE PROFILE

We have deployed the proposed ITI profile in a widely used modeling tool: *Sparx Systems' Enterprise Architect* [16] that supports the definition of profiles.

Figure 13 represents an ITI model produced with our deployed profile. This example provides a first evidence that our proposal reduces the ambiguity in modeling ITIs, while providing the recurrent ITI concepts used by ITI architects, such as data centers, servers, network types such as perimeter, intranet, extranet, firewalls, routers or switches. The increased preciseness facilitated by the use of a formal metamodel is rendered possible by specifying well-formedness rules upon it using OCL clauses.

 ISBN: 978-1-61208-230-1

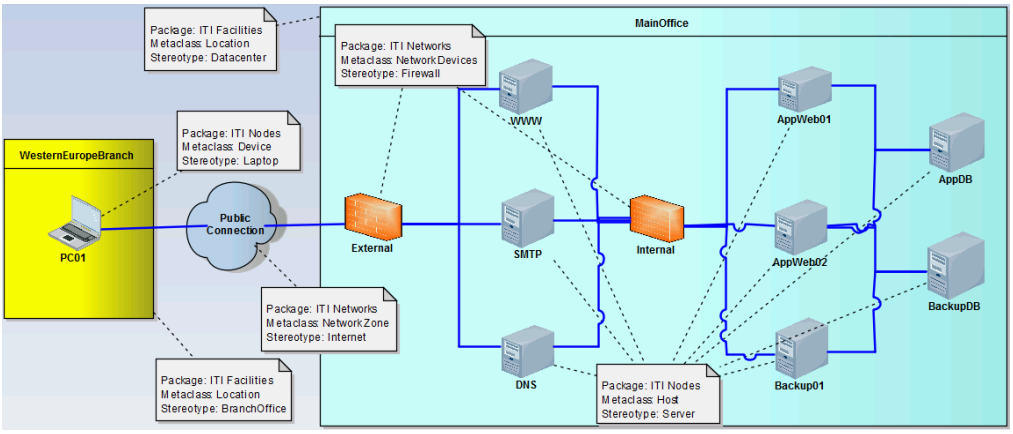

Figure 13. Simple ITI model using the proposed profile

V. CONCLUSION AND FUTURE WORK

This paper introduces a UML profile for modeling IT infrastructures, covering constructs at all required abstraction levels (hardware, middleware, network and software). Since the full profile has many stereotypes and each stereotype is described by means of several attributes, only a subset of the profile could be presented here. Nevertheless, we present some evidence that this profile mitigates the problems identified while reviewing existing approaches for modeling IT infrastructures:

i) The widely used ad-hoc approaches produce ambiguous models, do not facilitate knowledge reuse and cannot support validation approaches;

ii) The existing formal approaches for modeling ITIs, such as *UML Deployment Diagrams, TOGAF* or *ArchiMate*, do not provide the required abstractions and, probably due to that, are not used in practice.

Several future research threads relate to the availability of this profile:

ITI models capture – Organizations often have IT service management tools (e.g., CMDB – Configuration Management Data Base) that store information on the ITI elements. Being able to import that information and generate preliminary layouts is a major research concern.

ITI models scalability - Models of real-world infrastructures, even in medium-sized companies, can easily reach hundreds or even thousands of modeling elements, especially when software components are considered. In such a case, a model can easily be rendered useless due to excessive detail. We plan to mitigate this problem by using zooming facilities like those available in GIS.

Reuse ITI modeling best practices: We have proposed elsewhere the concept of *ITI patterns* [17, 18]. The availability of this profile will allow granting more preciseness to the formalization of those patterns.

ACKNOWLEDGMENT

This work is partly supported by grant PEst-OE/EEI/UI0527/2011 of Centro de Informática e Tecnologias da Informação (CITI/FCT/UNL).

REFERENCES

[1] A. Gunasekaran, H. J. Williams, and R. E. McGaughey, "Performance measurement and costing system in new enterprise," *Technovation,* vol. 25, pp. 523-533, 5// 2005.

[2] OGC, *IT Infrastructure Library (ITIL) - Service Design (Version 3)*. London: The Stationery Office, 2007.

[3] http://www.oracle11grelease2.com/services/infrastructure-velolux/.

[4] L. Ferreira da Silva, F. Brito e Abreu, and V. Moreira, "A UML Profile for Modeling IT infrastructures," presented at the INFORUM'2012, Caparica, Portugal, 2012.

[5] OMG, "Unified Modeling Language (UML) Specification: Superstructure (version 2.3)," ed, 2010.

[6] OMG, "Unified Modeling Language (UML) Specification : Infrastructure, version 2.3," 2010.

[7] A. Gerber, A. Van der Merwe, and P. Kotze, "Towards the Formalisation of the TOGAF Content Metamodel using Ontologies," presented at the Proceedings of the 12th International Conference on Enterprise Information Systems, Funchal, Madeira, Portugal, 2010.

[8] The Open Group, "Archimate 2.0 Specification," ed. Zaltbommel: Van Haren Publishing, 2012.

[9] T. Aldawud, A. Bader, and T. Elra, "UML profile for aspect-oriented software development," presented at the The Third International Workshop on Aspect-Oriented Modeling, 2003.

[10] T. Arpinen, T. Hamalainen, and M. Hannikainen, "Meta-Model and UML Profile for Requirements Management of Software and Embedded Systems," *EURASIP Journal on Embedded Systems,* 2011.

[11] B. List and B. Korherr, "A UML 2 Profile for Business Process Modelling," presented at the Perspectives in Conceptual Modeling (ER 2005 Workshop), Klagenfurt, Austria, 2005.

[12] OMG, "A UML Profile for MARTE: Modeling and Analysis of Real-Time Embedded Systems," vol. ptc/2008-06-09, ed: Object Management Group, 2008.

[13] U. Frank, D. Heise, H. Kattenstroth, D. Ferguson, E. Hadar, and M. Waschke, "ITML: A Domain-Specific Modeling Language for Supporting Business Driven IT Management," presented at the 9th OOPSLA workshop on domain-specific modeling (DSM), Helsinki, Finland, 2009.

[14] DMTF, "Common Information Model (CIM) Infrastructure," November 2007.

[15] http://ctp.di.fct.unl.pt/QUASAR *(accessed in 5/9/2012)*.

[16] http://www.sparxsystems.com *(accessed in 5/9/2012)*.

[17] L. Ferreira da Silva and F. Brito e Abreu, "Software distribution to remote locations," presented at the Proceedings of the 15th European Conference on Pattern Languages of Programs, Irsee, Germany, 2010.

[18] L. Ferreira da Silva and F. Brito e Abreu, "An IT Infrastructure Patterns Approach to Improve IT Service Management Quality," presented at the 7th International Conference on the Quality of Information and Communications Technology (QUATIC'2010), Porto, Portugal, 2010.

 ISBN: 978-1-61208-230-1