Towards a robust solution in Building Automation Systems: supporting rapid prototyping and analysis

Vasco Amaral
CITI, Departamento de Informática,
Faculdade de Ciencias e Tecnologia
Universidade Nova de Lisboa, Portugal
Email: vma@fct.unl.pt

Bruno Barroca
CITI, Departamento de Informática,
Faculdade de Ciencias e Tecnologia
Universidade Nova de Lisboa, Portugal
Email: mailbrunob@gmail.com

Paulo Carreira
IST and INESC-ID Lisboa
Taguspark Campus
Porto Salvo, Portugal
Email: paulo.carreira@ist.utl.pt

Abstract—It is presently required agile and systematic solutions aiming at streamlining the development, maintenance and configuration of complex Building Automation Systems (BASs) in an energy aware manner. We aim at defining usable Domain Specific Languages (DSLs) using a Software Language Engineering (SLE), as systematic approach for language development, and develop the right tools for specifying the behavior of BASs components along with their energy-related requirements. The goal is to not only assist the systems engineers while rapid-prototyping/developing affordable, high-quality, energy-efficient (EE) BASs, but also to take advantage of high level abstractions, efficient special-purpose verification algorithms and analysis tools for early validation and verification, in order to promote Quality of the generated software products.

We are watching to the rise of Model-Driven Development as the pragmatic approach to build them, since this approach is based on the notion of explicit abstractions/models. This is achieved thanks to model transformations that, besides automatically translating any specification of a given language into other execution specifications, also allow us to derive analysis specifications.

The problem with the referred types of transformation purposes is that we cannot guarantee quality and coherence between the derived specifications into execution specifications and combine it with analysis specifications unless we can either make use of testing over the execution, with the problems already known, or have mechanisms to study the transformations. While this problem is motivated by the concrete need of developing BASs, we foresee that it can be of general application in SLE.

In this position paper we will give a state of the art in Building Automation and we give an overview of a possible solution that uses MDD and model transformations, in the context of a BAS solution, in order to check their correctness in w.r.t. the formal semantics of the languages used in the target platforms (i.e either execution or analysis).

Index Terms—Software Language Engineering, Model Transformations, Model Transformations Analysis, Model Checking, DSLTrans, Quality in MDD

I. INTRODUCTION

It is expected that in the coming years Smart Buildings (SBs) and Smart Homes (SHs) will play a key role in the low-carbon economy [1]. The heart of an SB (or an SH) is the BAS[2]: an automated control system that coordinates electric and mechanical devices in order to improve the comfort and safety of the occupants of the space.

However, the development and adequate commissioning of BASs is expensive [5], preventing their widespread adoption therefore hampering significant energy reductions.

The main factors that contribute to the high costs of development and maintenance of BAS are (i) the lack of a sound/integrated tool support, to deal with the complexities of the design and implementation of an EE BAS, and (ii) the lack of tools that empower the non-technical users to configure the systems behavior according to their particular requirements.

We vision to specify the behavior of BASs components using declarative Domain Specific Languages (DSLs), unlike existing languages for automation, which are imperative in nature. SLE is the systematic approach do develop languages, and the respective research community is promoting MDD techniques to be used in order to rapidly derive integrated development environments (IDEs), to support this DSL with edition, validation and simulation capabilities that are able to assist the systems engineers while rapid-prototyping/developing affordable, high-quality, EF BASs. Besides, thanks to the use of explicit specification models, these systems become certifiable and easier to analyse and validate. Using the same principles, we envision to derive tools that empower end-users to commission, in run-time, their energy-aware BASs. Our proposal is disruptive in the sense that it aims at developing EE BASs through a sound and principled engineering practice.

However, it is known that depending on the expressiveness of the used language, the size of the model’s analysis spaces can be unreasonable leading to what is called a combinatorial explosion of the symbolic state space during the analysis of a given specification. This is often the case while analysing source-code expressed in typical programming languages used nowadays in the software industry.

Depending on the properties that we intend to check in our DSL’s specifications, we might have to answer to these two questions: (i) what is the most effective analysis tool to analyse such kind of properties; and (ii) our DSL should be perhaps less expressive, hence more restricted, in order to guarantee certain properties — either by construction, or at least ease the dynamic analysis of others.

Within this context of research, we have been developing a method, depicted in Figure 1, where we are able to formally guarantee that (i) all specifications expressed in a given DSL are correctly translated automatically towards the execution platform; and (ii) all specifications expressed in a given DSL...
are correctly translated automatically towards the analysis platform. Of course, in this case, correctness also means that we should ensure that the analysis results from each specification is coherent and sound w.r.t. its execution in the execution platform.

During this research we devised a syntax-to-syntax model transformation language designed to define analyzable language translations called DSLTrans [9]. This language imposes, by construction in its semantics, that all translations are confluent and terminating [10]. As shown in Figure 1, T1 and T2 are transformations specifications (e.g. can be expressed using DSLTrans’ Language) that drives the automated translation (the dark-grey arrows) of any specification of our DSL into both execution and analysis formalisms. Once we develop these translation specifications, we can then analyse them (depicted as the A light-grey arrows), and also validate them according to the formal semantics of both execution and target platforms.

II. RELATED WORK

Streamlining the development of Building Automation Systems is a complex and pervasive issue [11] that has been targeted by computer scientists as well as industry consortia through standards, languages and their supporting development tools.

A. BUILDING INFORMATION AND ENERGY MODELS

In AEC industry, well known Building Information Models [12] such as IFC [13] are used to model, among other aspects, building envelope and installed equipment characteristics. IFC MVDs are model transformations used to derive simpler models aiming at ulterior analysis and validation energy simulation.

Specialized descriptions of building equipment have emerged such as AEX [14] and building control domain of IFCs (in addition to including HVAC, Lighting). Energy behavior models of buildings, HVAC and Illumination subsystems are by now well understood [16], [6]. However, to the best of our knowledge, the inference/derivation of such models automatically from a BAS’ specification has never been addressed in literature.

B. DEALING WITH SOFTWARE COMPLEXITY: MODEL-DRIVEN DEVELOPMENT

In this project we propose to tackle the problems of designing a DSL (which is able to specify the structure and behavior of energy aware BASs), and building tool support for it (i.e., an IDE), by taking a MDD approach. In its essence, MDDs goal is to tackle software/information and hardware/physical systems complexity by modelling everything explicitly, at the most appropriate level(s) of abstraction using the most appropriate formalism(s) (DSLs) and simulating/deriving the target systems automatically. In order to translate these models into lower level abstraction languages, MDD provides Model Transformations, that can be either Model-to-Model or Model-to-Code. Currently, there exists a wide range of MDD tools for supporting Language implementation (also known as language workbenches) [17] that become specialized in the rapid prototyping of textual and graphical/diagrammatic editors for DSLs [18] (such as DSLTools, GMF, EMFText, among others), and in the specification and maintenance of the code generation and model transformation activities (such as QVT, Declarative QVT, VIATRA2, ATL, among others). Some transformation tools [10], [22], [23], [24] can even assure the quality of the transformations. For instance, some of them [10] assure that (i) every transformation will always terminates and give an unique result; and (ii) a given kind of properties is preserved in the transformation. An important evidence that the integration of these tools in a sound MDD methodology can actually deal with the increasing softwares complexity was realized with the BATICS project [25]. This project explored the MDD advantages in what concerns to usability and model verification (by means of model checking [26]),and DSLs extensibility (reusing platform independent model transformations).

C. APPLICATION OF MDD IN BUILDING AND INDUSTRIAL AUTOMATION

Currently, BAS are developed with platform- or vendor-specific tools like, ETS 3 or LonMaker, that do not place much effort in raising the level of abstraction. Many do not provide any type of concrete syntax, which is essential for the productivity in large projects, and when they do, their syntax is seldom intuitive and requires specialized training.

In view of the current state-of-the-art in MDD, we say that BAS are still developed using low level abstraction languages and lack a principled development methodology (as enabled by MDD). However there is some evidence that the BA and other related domains are gradually putting some of these MDD techniques into practice. For instance, techniques such as generative programming for assisting the creation of building
automation software [28] are examples of similar approaches, as well as model driven engineering [29].

Languages based in IEC 61131 have become a standard, recently superseded by IEC 61499 [36] to which much research effort has been directed concerning the application of MDD techniques for synthesis, validation and verification [37]. We champion using similar techniques for BAS.

III. CONCLUSIONS

We presently watch to a growing need for robust software products for the control of BAS with energy efficiency concerns. We propose to devise a framework that comprehends a DSL for restricting the design space while prototyping solutions. We also propose MDD as an enabler of such solutions. However, this solution alone is not enough since we still have the problem of not being sure about the quality of the obtained products because of the uncertainty about correspondence of what was modeled, analyzed and generated for execution regarding specific properties (e.g. energy efficiency). In this paper we gave an overview vision of what could be a possible solution based on research previously undertaken by us.

In the sequence of this dicussion we are undertaking a research plan where we i) build an appropriate DSL for Building Automation, that will lie on top of an existing Building Automation Platform; ii) choose and integrate the right formalism/tool to perform the analysis of this DSL’s specifications; iii) and provide a method (or a tool) to help us prove that the analysis over our DSL specification is sound w.r.t. the Building Automation Platform’s semantics. We expect that the results of the research in this Domain will provide us in the future with pragmatic means, formally grounded, to enforce robustness in MDD during SLE and therefore promote Quality in software products.

ACKNOWLEDGMENT

This work was developed in the context of the following research institutions: INESC-ID and CITI fund PEst-OE/EEI/UI0527/2011 Centro de Informtica e Tecnologias da Informao (CITI/FCT/UNL) - 2011-2012

REFERENCES


