Automation in Construction xxx (2016) xxx–xxx

Contents lists available at ScienceDirect

Automation in Construction

journal homepage: www.elsevier.com/locate/autcon

The Extensible Orchestration Framework approach to collaborative design in architectural, urban and construction engineering

Ana Perisic *, Marko Lazic, Branko Perisic

University of Novi Sad, Faculty of Technical Sciences, Trg D. Obradovica 6, 21000 Novi Sad, Serbia

ARTICLE INFO

Article history:
Received 7 January 2015
Received in revised form 25 July 2016
Accepted 12 August 2016
Available online xxxx

Keywords:
Architectural design
Building information modeling
Computer-supported cooperative work
Construction engineering
Domain analysis
Domain-specific modeling
Frameworks
Model-based simulations
Software tools interoperability
Urban planning

ABSTRACT

The proliferation of information technologies and the diversity of problem domains that heavily rely on software tool applications promote computer-supported cooperative work as a challenging discipline that drives the development process of contemporary and future engineering methods, standards, and tools. Consequently, a particular domain of expertise in engineering and scientific fields has emerged, demanding more advanced skills and deeper domain knowledge. The essential role of Architectural Design (AD) and Urban Planning (UP) is to enable a forward-looking approach to building/facility creation. Construction Engineering (CE) expresses its routine primarily through a transition phase that transforms ideas to sustainable urban artifacts. The CE role appears as a combination of backward and downward looking to the same process/product. All three domains are highly cooperative in the context of Environmental Engineering (EE). Currently available software tools that support the AD, UP and CE domains are far from simple. Several recent software engineering studies suggest that, instead of developing a complex "all in one solution", the federation or orchestration of several related simple methods and tools seems promising. In this article, we discuss the basic AD, UP and CE domain-cooperation aspects and suggest an Extensible Orchestration Framework (ExOF) model that may support them. To verify the ExOF simulation and orchestration potential, we used its architectural model to orchestrate different software tools when performing the urban blocks daylight illumination simulation for different urban block morphology models. For the exact simulation, we used the normalized sun-exposition data for the city of Novi Sad, Serbia. Also presented is an illustration of the methodology, modeling and orchestration potential of ExOF for the selected case study, together with the results, obtained for typical 3D models of selected urban block morphology patterns.

1. Introduction – research motivation factors

The complexity of today's engineering problems demands inter-domain and cross-domain cooperation of different experts through intensive teamwork. Thus, Computer-Supported Cooperative Work (CSCW) is a challenging discipline that drives the development process of contemporary and future engineering methods, standards, and computer-based systems. From embedded system design to enterprise architecture modeling, any domain expert faces the same situation: a multiplicity of views, a multiplicity of concerns, a multiplicity of models and heterogeneity of modeling artifacts [22]. With respect to domain-specific software tools design, domain experts and software tool designers are usually faced with a highly conflicted decision-making process concerning the exact definition of the problem domain; domain expert mental models; the intellectual clarity of fundamental concepts; the elegance and understandability of current domain methods, techniques, and tools; the structure and behavior of any novel tool or methodology that must be developed; and the efficiency and effectiveness of proposed solutions [34].

Both researchers and IT practitioners agree that the most difficult step in any engineering project is a Domain Analysis. A Domain is often understood as a family of systems exhibiting similar static structure, dynamic behavior and/or external functionality. The creation of a domain mental model based on empathy with domain experts may be a suitable means to gain mutual understanding among domain experts and software tool designers. Domain mental models capture a deep understanding of people's motivations and thought processes along with the emotional and philosophical context in which they are operating. Empathy with a person is distinct from studying how a person "uses" something. It extends to knowing what the person "wants to accomplish" regardless of whether he/she is aware of the "thing" that is being designed [72].

A common approach to initial domain foundation is to develop a Domain Ontology (DO) whose main role is to capture useful domain semantics and to describe its characteristics as a set of domain rules and functional dependencies [40]. The fundamental question is, how can

* Corresponding author.
E-mail addresses: anaperisic@uns.ac.rs (A. Perisic), arhitektura@live.com (M. Lazic), perisic@uns.ac.rs (B. Perisic).

http://dx.doi.org/10.1016/j.autcon.2016.08.005
0926-5805/

one acquire sufficient domain knowledge to formulate operationally usable abstractions? This question is particularly relevant with respect to the complex problem domains, in which inherent individual complexity may often discourage the software designer's efforts.

This inherent complexity, which is embedded in real-world concepts, promotes modeling and simulation as the unavoidable mechanisms of predictive and/or corrective engineering activities. In Sokolowski and Banks [53], "Model" is broadly defined as a formal or informal representation and abstraction of anything (a real system, a proposed system, a futuristic system design, an entity, a phenomenon, or an idea), whereas "Simulation" is defined as the act of executing, experimenting with, or exercising a model or a set of models to achieve specific objectives. A strong interdependency between Modeling and Simulation leads to the concept of Model-Based Simulations (MBSs).

Concerning the application of modeling concepts to contemporary software engineering, there are two main research directions that may be followed. The first one, the Object Management Group's (OMG) Model-Driven Architecture (MDA), emphasizes the importance of a single and universal modeling language foundation. The second one, Domain Specific Modeling (DSM), states that flexibility and ease of use are much more important than adhering to a single formalism. In Rath [44], the author defines Domain-Specific Modeling (DSM) as a top-down and vertical approach that, instead of trying to create high abstraction level "interfaces" to the implementation platform, provides domain experts the freedom to use problem domain specific structures and logic in a completely implementation-independent manner.

The recent explosion of Domain-Specific Language (DSL) development has established a challenging target for a novel "silver bullet" that drives the majority of current Model-Driven Engineering (MDE) studies. Focusing on language development and a language-oriented paradigm, this approach favors the development of dedicated languages that suit domain experts better, enabling them to specify concrete domain solutions by simply combining the highest possible common and standard abstract concepts taken from the particular problem domain. In Hudak [25], the author describes domain-specific languages (DSLs) as "small programming languages tailored for a particular application domain", consistently referring to families of specific, similar problems suitable for linguistic description. As such, DSLs can be viewed as sets of general, all-encompassing solutions for problem domains. Up-to-date approaches to Domain-Specific Language foundation building rely on [8,11,17,24,28–30,37,54].

Recent studies suggest the federation or orchestration of several related methods or tools in contrast to developing just one (an "all in one solution").The creation of DSLs is today much better supported than it was just a few years ago ([13,14,19]). For creating DSLs, the cited studies suggest the Language Workbench approach, which is based on the synergism of a relatively large number of simpler, dedicated tools and languages.

The application of previously mentioned paradigms in complex problem domains currently is, and will certainly remain, the research frame for the majority of system and software engineering activities. In our recent studies, we have focused on the cooperativeness of standards, methods, and tools in the Architectural Design (AD), Urban Planning (UP) and Construction Engineering (CE) domains of expertise.

Due to the proliferation of sophisticated information and communication technologies in these three domains, current engineering is difficult to imagine without intensive utilization of software tools. Even when supporting individual aspects of AD, UP and CE domains, up-to-date software tools are much more complex than tools available just a few years ago. The main challenge is that the tools constantly tend toward increasing complexity. To use even a fraction of their functionality efficiently, the development of sophisticated domain expert skills is essential. Unfortunately, such development usually takes a long time and demands a high degree of tool dependence, the major obstacle with respect to current software tools' development speed.

DSLs have already been quite successfully used in computer science, primarily to build intuitive user interfaces that are understandable by non-IT experts. However, until now, they have not been adequately applied to the areas of AD, UP and CE. In Kramer et al. [31], the authors suggest the intuitive interfaces (i.e., DSLs) as key factors supporting the participatory, bottom up, approach to urban planning. As an extension of this work, Kramer [32] presents a concrete DSL application in an interpreter-based user interface development for accessing Cloud-resident large geospatial data.

Model-Based Simulations (MBSs) are performed either to exercise the future behavior of an engineered artifact or to establish directives for potential revitalization of existing artifacts. For AD, UP and CE domains, special MBS challenges lie in the modeling and parametric simulation of the embedded features of cooperatively developed final product(s) (individual building(s), urban block(s) or a complete city(s)).

Particular solutions require appropriate general answers for the following issues: coping with future complexity in AD, UP and CE domains and incorporating MBSs in the engineering and reengineering of their artifacts. Raising the level of abstraction and the degree of artifacts' reusability are certainly promising approaches. Both are achievable through the development and utilization of a sustainable, cooperative framework that, from the domain expert's point of view, appears similar to a hyper-tool that facilitates a focus on problem domain-specific abstractions rather than on tool-dependent activities.

In accordance with the System of Systems formulation [50], we refer to this approach as the Tool of Tools (ToTs). ToTs development ultimately demands the existence of a multi-level common ontology developed on top of cooperative and collaborative potential of individual problem domains; the possibility of discovering the integrability potential of component tools (CT); a higher level of integrability support that must be embedded in future software tools, enabling their easier integration into arbitrary cooperative/collaborative associations; complexity-hiding mechanisms that utilize DSLs to formulate concrete orchestration models; and an extendibility mechanism based on strict separation of abstractions and their implementations and that aids the longevity of the proposed solution.

State-of-the-art AD, UP and CE exhibit little documented experience with Domain Specific Languages (DSLs) or Software Tools (SWT) orchestration frameworks, making our approach a challenging one.

2. Proposed solution fundamentals – domain-specific, framework-based integrability, extensibility and orchestration support

Our recent studies are particularly focused on the general, cross-domain commonalities of AD, UP and CE domains that must be included in a sustainable and usable cooperative framework. To aid the development of complex architectural/urban artifacts, a framework must support the cooperation of DSLs, Stand Alone Software Tools, and Industry Standards. With respect to these requirements, Multi-Domain Systems Engineering (MDSE) appears a suitable approach to the Extendable Orchestration Framework (ExOF) formulation.

2.1. Domain-specific foundation

The enormous research background may be referenced to justify the novelty of any approach concerning AD, UP and CE domains due to their historical position in overall civil and environmental engineering. Contemporary computer-supported cooperative tools are primarily designed to support domain experts in their everyday activities. Although domain experts are concerned, the soundness of a tool with a domain-dependent routine is based on the essential usability factor. This soundness ultimately requires that the tool's internal functionality must be wrapped by a user's interface that mimics the "hand-practiced

Please cite this article as: A. Perisic, et al., The Extensible Orchestration Framework approach to collaborative design in architectural, urban and construction engineering, Automation in Construction (2016), http://dx.doi.org/10.1016/j.autcon.2016.08.005