Intelligent Computing for Building Performance Analysis

Pieter de Wilde

Plymouth University, Plymouth, PL4 8AA, UK
pieter.dewilde@plymouth.ac.uk

Abstract. A challenge towards the intelligent use of computing in civil and architectural engineering is the definition of the questions that the ICT technology has to address. To some extent this is implicitly covered by activities such as the definition of search and option spaces, development of model views, the specification of objective functions, definitions of ontologies, or the development of multi-criterion decision methods. However, the underlying needs and drivers of design, construction and facility management processes of buildings are hard to capture, while they are essential to effective use of computing techniques. This paper reviews the starting point for intelligent computing within the domain of building performance analysis. It explores how approaches from the field of requirement engineering may help to support proper definition of computational needs, while embedding computational analysis efforts within the wider context of assessment approaches that are available in the building domain.

Keywords: Building performance · Requirement engineering
Intelligent computing

1 Introduction

The field of engineering informatics covers, amongst others, the use of computing technology in the building and construction discipline. A solid body of knowledge about engineering informatics has been developed, as represented by articles in peer-reviewed academic journals such as *Advanced Engineering Informatics*, books like the *BIM Handbook* [1] or *Fundamentals of Computer-Aided Engineering* [2] and, indeed, the proceedings of the annual workshop of the European Group for Intelligent Computing in Engineering (EG-ICE). Topics covered within engineering informatics cover issues such as data management, optimization and search, visualization, machine learning, (webbased) collaboration, building information modelling, and many others.

A challenge to intelligent use of computing in civil and architectural engineering is the definition of the questions that the ICT technology has to address. These questions are the true drivers of the computing effort and are highly important in discerning between intelligent and not-so-intelligent use of ICT. However, their definition is often left implicit. To some extent they are covered by activities such as the definition of search and option spaces, development of model views, the specification of objective functions, definitions of ontologies, or the development of multi-criterion decision methods. But the underlying needs and drivers of design, construction and facility management processes of buildings are hard to capture, while they are essential to effective use of computing approaches.

Building Performance Analysis is a wide field that deals with three constituent parts: an engineering view that explorers how well buildings meet functional requirements, a process view that studies the performance of the building construction process, and an aesthetic view that covers the architectural performance of buildings. In further detail, the engineering view deals with building quality, workload capacity, resource saving, timeliness and responsiveness [3]. Typical performance aspects of interest include structural stability, processing capacity, fire safety, energy efficiency, thermal comfort, lighting levels, acoustical comfort, and indoor environmental quality. The activity of building performance analysis takes place along the full building design life cycle, from initial definition of need, via design, actual construction and commissioning, management and operation, refurbishment and retrofit, to end-of-life disposal of buildings and their constituent parts. It brings together a wide array of stakeholders, including client, architect, civil engineers with a range of specialisms, contractors, facility managers and others. Combined with the fact that most buildings are complex, bespoke products that represent a system of systems, and that many performance aspects interact, this creates a challenging situation where the starting point for computational analysis effort is highly unique.

The aim of this paper is to explore the definition of underlying questions and drivers that from the starting point for intelligent computing within the domain of building performance analysis. The following objectives have been identified:

Objective 1: review the background of current building performance analysis software and systems;
Objective 2: investigate structured software analysis approaches from other domains;
Objective 3: develop suggestions on how drivers for building performance analysis may be captured, and explore how this may support the evolution of new approaches in the computational assessment of building performance;

The paper is builds on an extensive review of literature on building performance analysis that has been conducted in the context of a forthcoming book on the subject of Building Performance Analysis [4], but now taking the subject specifically into the computing domain and this extending the scope of the discussion. It positions intelligent computing of building performance in a wider context that compares and contrast computational analysis – mainly in the guise of building performance simulation – with other assessment approaches such as physical measurement, stakeholder assessment, and expert judgment.

2 Background on Building Performance Analysis Software Systems Development

Building performance analysis has a long history. Even in primitive shelters as constructed by early humans performance aspects like protecting occupants from the elements and providing safety from wild animals plays a key role. Initially performance of buildings will have been explored through trial and error. Later it became the domain of ‘master builders’ and architects. A seminal contribution to the field are the books on architecture by the Roman architect Vitruvius – also a civil engineer – who considered that buildings must possess three key qualities: *firmitas, utilitas* and *venustas*, roughly translated as strength, utility and beauty [5]. Further disciplines within the building domain emerged during the industrial revolution, which saw the development of specialisms like structural engineering and building services engineering. Interestingly, this still seems to impact the situation today. While different aspects all fit the notion of building performance, there still is a clear split between the domain of structural engineering and a different ‘blood group’ of aspects that are clustered as ‘building science’; building science typically covers heat and mass transfer, lighting, acoustics, and indoor air quality. A further body of knowledge on building performance was established in the late 1960s to early 1980s, spearheaded by the work of the Building Performance Research Unit (BPRU) at the University of Strathclyde and the CIB Working Commission W60, with the latter focussing on ‘working with the performance approach to building’. Pressures from issues like sustainability, climate change, limited resources, health and safety and occupant wellbeing keep moving the field forward. However, it must be noted that thermal aspects (energy use, thermal comfort) seem to be rather dominant in the computing side of the building science domain, followed by some interest in lighting and acoustics. Other aspects such as building evacuation modeling are niche areas, while some areas like burglary resistance lack meaningful simulation approaches.

Computing has always played an important role in quantifying the performance of a building for the key performance aspects. The introduction of desktop computing in the mid 1970s saw a step change in possibilities. This led to the emergence of a new domain named building performance simulation. For the history of this field see for instance the descriptions by Augenbroe [6] or Clarke [7]; for the detailed history of a selection of whole-building thermal simulation tools see for instance the work by Oh and Haberl [8]. It is interesting to note that the building science area has tended to develop ‘closed tool boxes’ in the form of programs like DOE-2, ESP-r, TRNSYS and EnergyPlus, while the structural engineering area has shown a tendency to stay closer to general engineering approaches and underlying mathematical equations as available in programs like Matlab, ANSYS and Mathematica.

Evolution of building performance simulation tools is a slow process. Many tools used today have been in existence for years: TRNSYS dates back to 1973, DOE-2 to 1975, and ESP-r to 1974. EnergyPlus was launched in its first version as recent as 1997 but includes a legacy of DOE-2 and BLAST. Even with regular updates, as are provided for EnergyPlus, this means that many of the underlying assumptions and computational routines have been around for a long time. For comparison, the more general tools have similar histories: ANSYS was first released in 1971, Matlab in 1984, and Mathematica in 1988.

A lot of recent efforts in development of tools for building performance analysis seems to be invested in building shells around existent building performance analysis engines, such as DesignBuilder, Safaira and OpenStudio environments around EnergyPlus, or IES around Apache.

The evolution of building performance analysis software cannot be seen isolated from the development of Building Information Models or BIM. Seminal work in the area as described by Eastman [9] shows how BIM systems emerged partly as a result of the desire to share data amongst various computer applications, in what was then known as product models. Work like the EU Combine Project [10] shows that these efforts specifically aimed for performance areas such as building energy efficiency, costs and aesthetics. Over the years, two types of developments can be observed: the development of tool-independent BIM infrastructure, such as the BuildingSmart International Foundation Classes (IFC), and the development of relatively 'closed' suites of interoperable tools such as those connected within the IES Software system. Obviously, the tool-independent infrastructure allows for flexibility and thus sees wider application, whereas the closed interoperable systems have a more constrained use domain. However, the tool-independent approach tends to suffer from information overload. This has led to the development of domain-specific filters, named Model View Definitions (MVDs). While MVDs are a step towards better management, they are not yet perfect; Lee et al. discuss the challenges in defining MVDs and how these issues may lead to inconsistencies in exchange specifications [11]. An overview of some of the practical issues when using BIM in a multi-disciplinary collaboration is provided by Singh et al. [12]. The 'closed' suites like DesignBuilder or IES limit the number of interactions and hence are less prone to data exchange problems, but this comes at the price of reduced flexibility. However, IES at the moment seems to be a well-accepted industry standard in the building services engineering sector; both IES and DesignBuilder enjoy a wide uptake in the education sector.

While the literature offers a wide description of technical details of building performance analysis tools – for instance the EnergyPlus Engineering Reference document alone is 847 pages long – there is a sparsity of information on the user needs that these tools aim to address. There are several academic papers, such as Petersen and Svendsen [13] or Negendahl [14] that describe the needs that tools need to respond to, but these do not provide insights into the requirements that drive actual tool development by the major software houses and academic communities. Papers from tool developers tend to focus on the features that their tools provide, rather than on the underlying requirements.

A general understanding of what existing tools respond to can be achieved by looking at the various tool capabilities that are used to define tool categories in the Building Energy Software Tool Directory (maintained by IBPSA-USA), available from www.buildingenergysoftwaretools.com, as presented in Table 1.

The following observations can be made with regards to these categories. First of all, while the name of the directory singles out the single performance aspect of energy, the list includes tools that deal with other aspects such as lighting and water. Secondly, while most categories relate to professional building design and facility management, training is recognized as a separate need. Thirdly, the categories of calibration and weather data analysis in fact deal with modeling efforts, rather than straight building performance analysis tasks. Fourth these categories mix analysis activities, such as load prediction and auditing, with a building system typology, such as the split between envelope and HVAC systems.

Another interesting issue is that of the intended users of building performance analysis software. The Building Energy Software Tool Directory lists the target "audience" for each individual tool, naming for instance: architects, architectural designers, architectural engineers, builders, building energy modelers, consultants, contractors, daylighting designers, design evaluators, educators, energy code writers, energy managers, engineers, homeowners, HVAC designers, lighting designers, managers, manufacturers, mechanical engineers, policy makers, professionals, researchers, simulation experts, students, sustainable design engineers, tenants and urban designers. Obviously this is a very wide range of tool users, with a strong variation in background in terms of proficiency in using computational tools as well as training in the principles that underlie building performance analysis.

While the academic literature only offers a limited insight in the user needs that the existent tools try to meet, a generally accepted approach appears to be to use BIM systems to define building properties, pushing data from the BIM to a range of analysis tools, and then proceeding with specific evaluations. One of the many examples that describes this approach is the work by Oduyemi and Okoroh [15], which lists the following steps being required: (i) description of the site (ii) description of the building (iii) selection of relevant 'design indicators' (iv) development of baseline performance levels (v) exploration of '*what-if*' scenarios which focus on specific interest, such as system and operational parameters and (vi) uncertainty and sensitivity analysis. The notion of '*what-if analysis*' seems to be a wider trend in the industry, and rests on the premise that the best way to support building performance analysis is to build a model of a building, and then explore the impact of the variation of properties and parameters; see for instance Hopfe and Hensen [16]. More advanced approaches are available from the domain of statistics, where the theory of 'design of experiments' (DOE) employs the principles of randomization, replication and blocking in order to allow for factorial experimentation with efficient evaluation of the variation across a set of different parameters [17]. This approach can become more demanding in situations where the analyst may want to explore different system configurations, such as in the case of selection of HVAC system components, or building retrofit: in some cases there may be pre-configured system models that can be switched on and off, but in cases where a system needs to be introduced from scratch it may require a significant modelling effort.

Within the building performance analysis field, and especially those sub-areas that are concerned with environmental and sustainability issues, there is special attention for the use of computational tools during design. The use of tools to support design decisions has important advantages: since the building is not yet in existence, this is the only way to predict the performance of what during design are merely plans. It allows to subject design proposals to exactly identical testing conditions, something that is often very hard to do in real life buildings, where the best one may achieve is a semi-controlled experiment, since one has only limited control over things like occupant behaviour and climate conditions.

There is deep debate how to best support building design, with some authors emphasizing the need to equip designers with tacit knowledge [18] while others suggest further

Table 1. Tool capability categories used by the building energy software tools directory.

Whole-building energy simulation	Building energy benchmarking
Load calculations	Lighting simulation
HVAC system selection and sizing	Indoor air quality simulation
Parametrics and optimization	Life-cycle analysis
Model input calibration	Detailed envelope simulation
Energy conservation measures	Detailed component simulation
Code compliance	Solar and photovoltaic analysis
Ratings and certificates	Electrical system simulation
Utility bill and meter data analysis	Water use analysis
Weather data and climate analysis	Training services
Building energy auditing	Other

rationalization of the design process and stress the importance of design decisions that are underpinned with evidence [19]. Further attention is directed to attempts to match computational efforts to characteristics of the design process, leading to a longstanding and persistent claim that tools need to support fast design evolution and permutation [20]. Similarly, there is significant discussion about the need to support early design decisions; it is generally believed that early design decisions may have more leverage on achievement of building performance, whereas later decisions concern a building design that is already 'locked in place' and thus have less impact. This conflicts with the amount of detail that may be used in the evaluation of performance, leading to what is commonly known as the 'design paradox' [21, 22]. Further work promotes the combination of building performance analysis tools with optimization algorithms as a way forward to arrive at optimal design solutions [23, 24].

While these are all serious issues, it leads to a situation where the criteria for performance analysis remain very generic. For instance, Attia et al. list the following issues as important in getting simulation tools integrated into the design process: (i) quick analysis that supports decision making (ii) incorporation of uncertainty and sensitivity analysis of key parameters (iii) capability for weather analysis and suggestion of appropriate solutions (iv) ability to be used across various design stages [25]. Such recommendations are useful at a holistic level, but are hardly appropriate to guide the creation of software in a way that can be validated and verified.

There are two efforts that strictly speaking are not drivers for software development, but which aim to provide a better fit between design activities and computational analysis efforts. However, these provide some interesting insights in the deeper requirements. The first is the notion of 'performance assessment methods' or PAMs, developed in the context of International Energy Agency Annex 21 which ran from 1988 to 1993. A PAM sets out the information needs for a particular building performance analysis effort, such as the analysis of overheating risk. The idea behind PAMs was to provide a tool-independent definition of analysis needs in order to enable comparison of calculation methods. In theory, this might also be used as a specification of requirements for tool development [26]. The second is the development of Analysis Functions (AFs) in the context of the Design Analysis Interface (DAI) Initiative. AFs are defined to enable an efficient data parsing from a central BIM model to dedicated performance analysis tools, providing a template of the information that is required to undertake a specific assessment. AFs are linked to tasks in a process that is modelled and enacted in a workflow management system; however the stakeholders and their activities used are rather traditional and based on academic insights rather than operational studies of actual needs [27].

This brief review of the background of building performance analysis software systems gives an overview of main developments and trends. Obviously there is more detail in various papers and handbooks that introduce the tools currently on the market. However, it seems safe to state that most building performance analysis tools have been developed iteratively, using a trail and error process, and that natural selection over a period of around 50 years has resulted in the emergence and retention of the present toolset. Current tools are still heavily reliant on legacy 'calculation engines', with the models embedded in many tools dating back to the 1970s; recent efforts seem to focus more on building shells around tools, with those shells offering advanced interfaces for handling building geometry, quick access to default systems and settings, and reducing modeling requirements – see for instance the efforts on DesignBuilder, Sefaira and OpenStudio around the EnergyPlus simulation engine, or IES around the legacy Apache engine. A special branch of software development is also emerging around the Rhinoceros 3D design application, where a set of add-on applications such as Grasshopper, Ladybug and Honeybee helps to interface with simulation engines like EnergyPlus and Radiance, as well as use generative functions.

3 Structured Approaches from Other Domains

While the development of building performance analysis software requires deep subject knowledge, efforts in this area sometimes become rather inward-looking, ignoring developments in the wider context. For instance there is a significant body of knowledge on Software Engineering, as exemplified by the seminal textbook by Pressman [28]. Typically, this stresses the fact that all software development takes place in response to some kind of business demand. Any programming efforts are preceded by identification of the stakeholders and their needs, followed by planning of the process, design of the system architecture, software construction/coding, testing and ultimately deployment. There are different processes, which can be highly linear and prescriptive, incremental and iterative, or evolutionary. Yet, as stated by Pressman: '*Understanding the requirements of a problem is among the most difficult tasks that face a software engineer..... even if customers and end-users are explicit in their needs, those needs will change throughout the project. Requirements engineering is hard.*' Yet this is the essential work that defines how software fits in a business process, meets the need of the client, and how end-users will interact with the product.

Software development can be seen as part of the wider realm of Systems Engineering [29]. Systems engineering is an interdisciplinary field of science that deals with the design and management of systems, where systems may be physical systems ('hardware'), IT systems ('software'), business and services. Like Pressman on software, the International Council on Systems Engineering (INCOSE) emphasizes the need to start with the customer, saying that systems engineering '*focusses on defining customer needs and required functionality early in the development cycle, documenting requirements, and then proceeding with design synthesis and system validation while considering the complete problem*' [ibid]. This requires the analysis of the business or mission, definition of stakeholder needs and requirements, and identification of system requirements. Gilb points out that a key challenge in project management and engineering efforts is that it is difficult to articulate requirements and to cope with changes in requirements that have been set [30]. He goes on to list the following list of key issues for proper definition of requirements:

- identification of critical stakeholders;
- separation of requirements from design ideas (keeping requirements and solutions apart);
- prioritization of key requirements that are critical for system success;
- definition of what will be considered system success, or system failure;
- comparison of requirements to benchmarks;
- development of timescales for delivery.

While these are relevant issues to keep in mind, it still leaves open how one actually does identify the stakeholder needs. This can be done through requirement engineering, the branch of systems engineering that aims to capture and describe the client needs and expectations for a new product. Requirement Engineering consist of the following fundamental activities [29]:

(1) requirement elicitation, the process of identifying stakeholders in a new product and their needs and desires;
(2) requirement documentation, which involves the description of the requirements stemming from the elicitation process in words and models;
(3) validation and negotiation, the process of checking that the requirements are complete, reflect true stakeholder needs, and solving any conflicts;
(4) requirement management, maintaining track of changes and ensuring consistency.

Further detail on requirements engineering can be found in the publications by Pohl and Rupp [31] or Robertson and Robertson [32]. Amongst others, these works suggest the use of various techniques like brainstorming, analysis of existing systems to identify stakeholders, workshops and interviews to elicit requirements from stakeholders, and expression of requirements by means of language template ('boilerplates') and formal modeling techniques. A special technique is the definition of 'use cases', which describe the detailed interaction between a user and a system. Use cases help to reduce the complexity of large systems, dividing the overall system functionality into smaller views that correspond with the activities that system users will undertake when employing the system. Modern visualization languages such as UML (Unified Modeling Language) and IDEF (Integration DEFintion) have dedicated diagrams that help to depict use cases.

Some further interesting observations can be made from within the construction/ building engineering domain. For instance, Lucas et al. demonstrate the use of UML Use Cases to explore the information needs of a healthcare facility, and how this may be used to design an IT system for facility management of a hospital [33]. Wang et al.

explore the anticipated activities of future building occupants and how this relates to the building lay-out using a BIM-based system; interestingly this is almost the development of use cases but for the building itself, not for the software used to support the design process [34]. While these efforts focus on facility design and management, it shows the applicability of UML Use Cases in the construction sector. Girodon et al. point out that software may help to automate repetitive design tasks, but that efficient systems should relate to expertise and knowledge of their users; they suggest an agent-based approach to tailor systems to specific users [35]. Their approach not only discerns different actors and activities but also different roles and missions. Chong and Chen discuss that stakeholder (customer) needs are not static, but may evolve and change; their customer requirement analysis and forecast (CRAF) system attempts to address this challenge [36]. Dynamic requirement development is definitely something one would expect for software engineering, where regular updates are now common place across most platforms. Wang et al. go on to explore how user requirements may change due to interactions between the product, user behaviour, motivation, and perceived value [34].

Golzarpoor et al. note that modern information systems combine data management with the application of efficient and effective processes. However, in building and construction the emphasis in IT systems is on product information; process control and workflow management receive only very limited attention [37]. By way of example Luo et al. present a system for developing and managing the functional performance specifications for a building, focussing on support of the briefing process rather than on requirements posed for the building performance analysis software [38]. A general structure for supporting IT-based collaboration between project partners is presented by Ren et al. However, the focus in this work is on supporting the planning process; it shows the complexity of the many interactions between various actors, roles and processes but does not delve into the specific tasks that are to be supported by building performance analysis software [39].

Most interest in building performance analysis computations goes towards quantitative assessment. Providing qualitative design support to architects and engineers however is not straightforward either due to the complexity of buildings, uncertainties, and information often being vague and incomplete at design stage [40]. However, the starting point for any intelligent computational effort should be a clear requirement definition. For instance, in mechanical engineering, design space exploration is explicitly linked to the definition of a system architecture. Gadeyne et al. [41] provide an overview of the description of the design space for gearbox design using the Object Constraint Language (OCL) and Systems Modeling Language (SysML). Even with a such a well-defined system, with limited degrees of freedom, capturing the design space is clearly non-trivial.

4 Definition and Modeling of Building Performance Analysis Drivers

Taking a steer from requirements engineering, this section provides an initial inroad into the definition and modeling of building performance analysis efforts. A first step is the exploration of stakeholders in computational analysis, and definition of use cases.

The range of stakeholders that have an interest in building performance is long. Many authors have provided overviews, listing clients, developers, building occupants or users, government at local and national level, society at large, architects, engineers, specialist consultants, contractors, product manufacturers, facility managers, financial institutes, insurers, and others. Amongst these, two groups get the most interest where it comes to definition of building performance analysis tools: architects and engineers; see for instance the papers by Bleil de Souza [18] and Attia [25]. However, it must be born in mind that these stakeholders are in fact categories and risk stereotyping. In practice buildings are mostly designed and engineered by many people with a wide range of personal traits, expertise, training and qualifications. In terms of qualifications one may discern between architects, architectural engineers, architectural technologists, mechanical engineers, building services engineers, construction managers, building science consultants, energy specialists, all with their own professional bodies (for instance in the UK these would be the likes of the Royal Institute of British Architects RIBA, Chartered Institute of Architectural Technologists CIAT, the Institution of Mechanical Engineers ImechE, Chartered Institution of Building Services Engineers CIBSE, or the Energy Institute IE). Even when using these formal qualifiers, real life will depend on personal interactions and dynamics; for instance Negendahl points out three main cases: (i) an architect working with an engineer as assistant (ii) a hybrid professional that combines both roles in one person and (iii) an architect and engineer working as equal partners [14]. Obviously this may be expanded with a case that fits situations where technology is dominant, such as chemical plants, where the case would be (iv) an engineering taking the main lead, with an architect as assistant. These same models can be applied to all other permutations of professionals, and expanded to teams that involve more than just two actors. Further complexity is added by the fact that these professional qualifications can come with different levels of training; for instance one may distinguish between novices, intermediate-level and experts in each category.

A proper identification of the many stakeholders in Building Performance Analysis however is just the first step in proper defining what is required of computational efforts. Further work is needed to identify which of these stakeholders, or what group of stakeholders, may be the software system user. This then can be followed up by an attempt to identify use cases. A first step in this direction that can be found in the literature are the building simulation use patterns as described by Tucker and Bleil de Souza [42]. This exploration of patterns could be expanded, empirically exploring the current activities of a range of AEC professional and how studying why and how they use existent systems. This would lead to a range of UML Use Case diagrams, as illustrated by Fig. 1, which is based on a theoretical range of software uses by an Architectural Engineer.

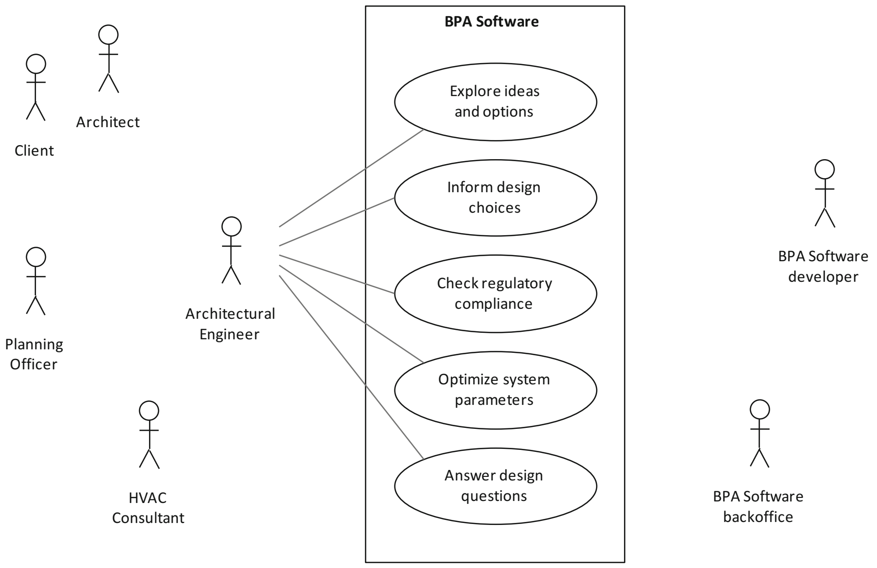

Fig. 1. Simple UML use case diagram depicting the use of building performance analysis software by an architectural engineer.

However, the situations depicted in Fig. 1 and in typical Use Case diagrams are only a first starting point for the interaction between any stakeholder and typical building performance analysis software. Further analysis will reveal a range of tasks and activities, any underlying process logic, and the interdependency of various data streams. Again as an illustration, Fig. 2 depicts a range of typical activities encountered when

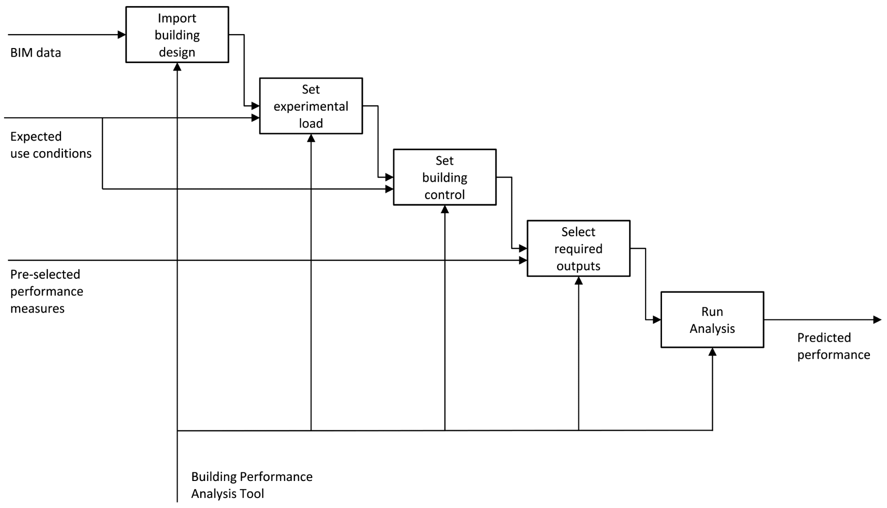

Fig. 2. Building performance analysis process logic (starting-point/use case dependent)

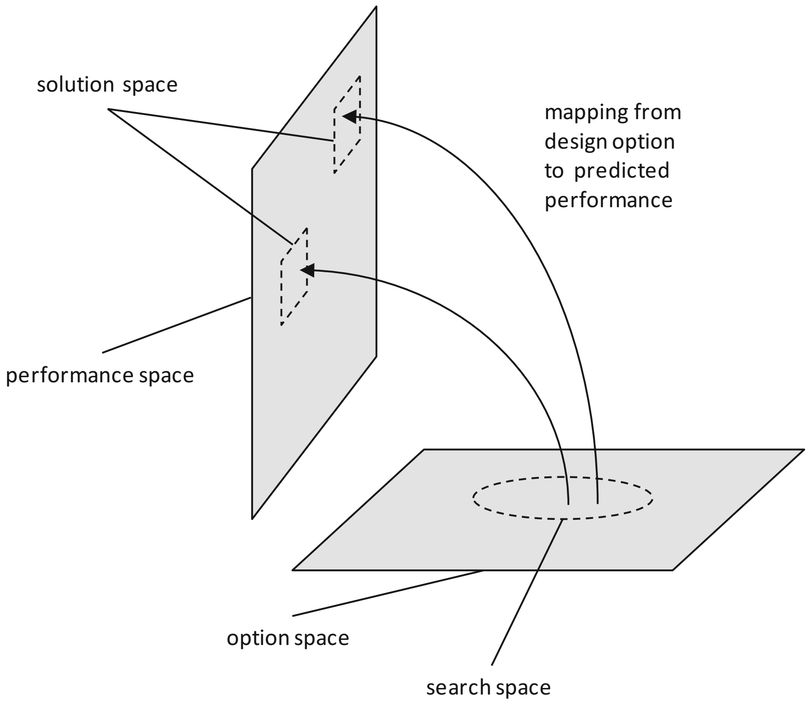

Fig. 3. Contextualization of mapping from search space to solution space.

using the current generation of software, such as EnergyPlus or DesignBuilder software. Note that this is only a crude attempt at high-level identification of tasks. Extensive empirical research would be needed to review actual workflows as occurring in the daily practice of various software users, after which categorization and standardization would provide realistic views in more detail. However, the depicted workflow already is helpful in defining various tasks needed to capitalize on the information that can be gained from a typical BIM system, as well as additional data needed to set up more specific analysis requests. At actual task level the workflow will be highly specific, providing a further focus for approaches that go further than the still quite generic Model View Definitions.

Beyond the workflow logic as captured in Fig. 2, further exploration is needed with respect to two terms that are used rather generally in literature on building performance analysis efforts: the 'search space' and 'solution space'. Typically these terms are encountered in the context of optimization. Search space is used to describe all system variants that are considered, whereas solution space is used to describe the corresponding predictions of system performance. However, further detail and discussion can be added. On the system side, one could make a difference between the 'option space' and the 'search space'. Here the option space would contain all the systems variants that are theoretically possible; the search space would be a subset of the option space, only containing those elements that are of interest to the stakeholder. Note that option spaces and search spaces may be continuous or discrete. For buildings, they are typically huge, with a large number of design parameters, system options, and attributes for each system. For each option, there then is a another space that contains the performance of that system option. Again, this is a multidimensional space, containing performance for aspects such as thermal comfort, energy efficiency, structural stability, visual comfort, and many others. As the list of performance aspects is long, it is unlikely that the whole performance space will be studied; instead, actual analysis efforts will focus on subsets which are limited by prediction capabilities and efforts, which can be designated as the solution space. The task of building performance analysis software then is to provide a mapping from search space(s) to solution space(s). See Fig. 3. Note that further research on option, search, performance and solution spaces is also required. On the system side, there are deep constraints in terms of system dimensional coordination, system compatibility, and standardization that render option and search spaces far from simple and continuous. On the performance and solution space side, single deterministic performance values are a typical oversimplification, and work is needed to incorporate various uncertainties and sensitivities in the mapping.

5 Discussion and Conclusion

This paper has provided an overview of the background of the software currently available for building performance analysis. It has also explored some of the approaches available in other domains, notable systems engineering and requirements engineering, which may help set more specific software development aims. This is followed by an exploration of how some concepts may be applied in the building performance analysis domain, and how these may lead to novel insights.

The paper concludes that intelligent computing of building performance has much to gain from investing time in proper development of analysis needs, in order to ensure that efforts are directed towards the things that really matter to the stakeholders. Rather than continue along a path of slow evolution, where the needs that drive the use of computational tools are left implicit, this approach would drive more specific and intelligent deployment of software. This may lead to a step change in the development of building performance analysis software that is more responsive to analysis needs, and is less dependent on tacit knowledge of software uses about the potential of their analysis tools. This paper present a first exploration towards further efforts in this direction.

References

1. Eastman, C., Teicholz, P., Sacks, R., Liston, K.: BIM Handbook – A Guide to Building Information Modelling, 2nd edn. Wiley, Hoboken (2011)
2. Raphael, B., Smith, I.: Fundamentals of Computer-Aided Engineering. Wiley, Chichester (2003)
3. de Wilde, P.: The concept of building performance in building performance simulation – a critical review. In: Barnaby, C., Wetter, M. (eds.) 15th International IBPSA Conference on Building Simulation 2017, San Francisco (2017)
4. de Wilde, P.: Building Performance Analysis. Wiley-Blackwell, Hoboken (2018)

5. Morgan, M.: Vitruvius: The Ten Books on Architecture. Dover Publications, New York (1960)
6. Augenbroe, G.: Trends in building simulation. In: Malkawi, A., Augenbroe, G. (eds.) Advanced Building Simulation. Spon Press, New York (2003)
7. Clarke, J.: Energy Simulation in Building Design, 2nd edn. Butterworth-Heinemann, Oxford (2001)
8. Oh, S., Haberl, J.: Origins of analysis methods used to design high-performance commercial buildings: whole-building energy simulation. Sci. Technol. Built Environ. **22**(1), 118–137 (2016)
9. Eastman, C.: Building Product Models: Computer Environments Supporting Design and Construction. CRC Press, Boca Raton (1999)
10. Augenbroe, G.: COMBINE 2 Final Report. Commission of the European Communities, Brussels (1995)
11. Lee, Y., Eastman, C., Solihin, W.: An ontology-based approach for developing data exchange requirements and model views of building information modeling. Adv. Eng. Inform. **30**, 354–367 (2016)
12. Singh, V., Gu, N., Wang, X.: A theoretical framework of a BIM-based multi-disciplinary collaboration platform. Autom. Constr. **20**, 134–144 (2011)
13. Petersen, S., Svendsen, S.: Method and simulation programs informed decisions in the early stages of building design. Energy Build. **42**, 1113–1119 (2010)
14. Negendahl, K.: Building performance simulation in the early design stage: an introduction to integrated dynamic models. Autom. Constr. **54**, 39–53 (2015)
15. Oduyemi, O., Okoroh, M.: Building performance modelling for sustainable building design. Int. J. Sustain. Built Environ. **5**, 461–469 (2016)
16. Hopfe, C., Hensen, J.: Uncertainty analysis in building performance simulation for design support. Energy Build. **43**, 2798–2805 (2011)
17. Montgomery, D.: Design and Analysis of Experiments, 8th edn. Wiley, Hoboken (2013)
18. de Souza, C.B.: Contrasting paradigms of design thinking: the building thermal simulation tool user vs the building designer. Autom. Constr. **22**, 112–122 (2012)
19. Becker, R.: Fundamentals of performance-based building design. Build. Simul. **1**(4), 356–371 (2008)
20. Marsh, A.: Peformance Analysis and Conceptual Design. Ph.D. thesis. University of Western Australia, Perth (1997)
21. Macdonald, I: Quantifying the Effects of Uncertainty in Building Simulation. Ph.D. thesis. University of Strathclyde, Glasgow (2002)
22. Marsh, R.: LCA profiles for building components: strategies for the early design process. Build. Res. Inf. **44**(4), 358–375 (2016)
23. Machairas, V., Tsangrassoulis, A., Axarli, K.: Algorithms for optimization of building design: a review. Renew. Sustain. Energy Rev. **31**, 101–112 (2014)
24. Nguyen, A., Reiter, S., Rigo, P.: A review on simulation-based optimization methods applied to building performance analysis. Appl. Energy **113**, 1043–1058 (2014)
25. Attia, S., Hensen, J., Beltrán, L., De Herde, A.: Selection criteria for building performance simulation tools: contrasting architects' and engineers' needs. J. Build. Perform. Simul. **5**(3), 155–169 (2012)
26. International Energy Agency: Annex 21 – Calculation of Energy and Environmental Performance of Buildings. Subtask B: Appropriate Use of Programs. Volume 1: Executive Summary. Building Research Establishment, Watford (1994)

27. Augenbroe, G., de Wilde, P., Moon, Y., Malkawi, A., Choudhary, R., Mahdavi, A., Brame, R.: Design Analysis Interface (DAI) Final Report. Georgia Institute of Technology, Atlanta (2003)
28. Pressman, R.: Software Engineering: A Practitioner's Approach, 6th edn. McGraw-Hill, New York (2005)
29. INCOSE: Systems Engineering Handbook: a Guide for System Life Cycle Processes and Activities. 3th edn. Wiley, Hoboken (2015)
30. Gilb, T.: Competitive Engineering: a Handbook for Systems Engineering, Requirements Engineering, and Software Engineering using Planguage. Butterworth-Heinemann, Oxford (2005)
31. Pohl, K., Rupp, C.: Requirements Engineering Fundamentals, 2nd edn. RockyNook, Santa Barbara (2015)
32. Robertson, S., Robertson, J.: Mastering the Requirements Process, 3rd edn. Pearson Education, Upper Saddle River (2012)
33. Lucas, J., Bulbul, T., Thabet, W.: An object-oriented model to support healthcare facility information management. Autom. Constr. **31**, 281–291 (2013)
34. Wang, Y., Yu, S., Xu, T.: A user requirement driven framework for collaborative design knowledge management. Adv. Eng. Inform. **33**, 16–28 (2017)
35. Girodon, J., Monticolo, D., Bonjour, E., Perrier, M.: An organizational approach to designing an intelligent knowledge-based system: application to the decision-making process in design projects. Adv. Eng. Inform. **29**, 696–713 (2015)
36. Chong, Y., Chen, C.: Management and forecast of dynamic customer needs: an artificial immune and neural systems approach. Adv. Eng. Inform. **24**, 96–106 (2010)
37. Golzarpoor, B., Haas, C., Rayside, D.: Improving process conformance with industry foundation processes (IFP). Adv. Eng. Inform. **30**, 143–156 (2016)
38. Luo, X., Shen, G., Fan, S.: A case-based reasoning system for using functional performance specification in the briefing process of building projects. Autom. Constr. **19**, 725–733 (2010)
39. Ren, Z., Anumba, C., Augenbroe, G., Hassan, T.: A functional architecture of an e-Engineering hub. Autom. Constr. **17**, 930–939 (2008)
40. Schulz, C., Amor, R., Lobb, B., Guesgen, H.: Qualitative design support for engineering and architecture. Adv. Eng. Inform. **23**, 68–80 (2009)
41. Gadeyne, K., Pinte, G., Berx, K.: Describing the design space of mechanical computational design synthesis problems. Adv. Eng. Inform. **28**, 198–207 (2014)
42. Tucker, S., Bleil de Souza, C.: Thermal simulation outputs: exploring the concept of patterns in design decision making. J. Build. Perform. Simul. **9**(1), 30–49 (2016)