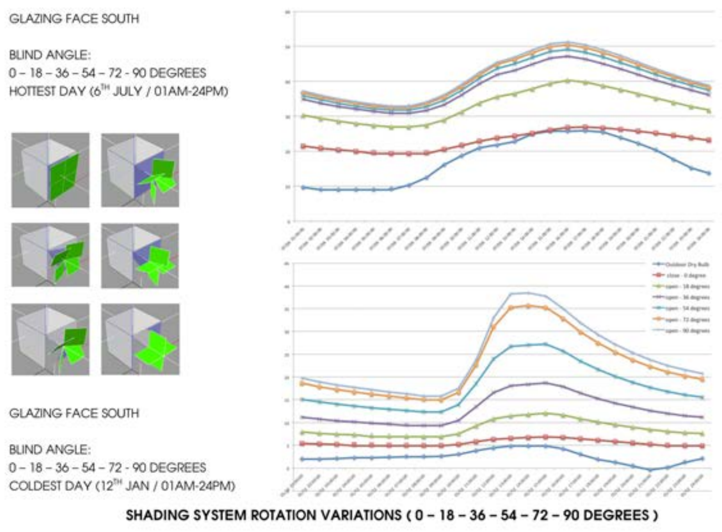

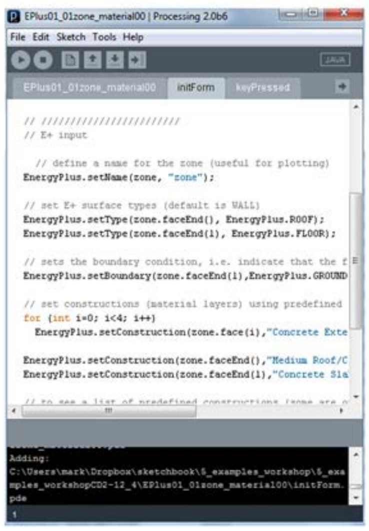
EPlus01_01zone_material00 | Processing 2.0b6
File Edit Sketch Tools Help
JAVA
EPlus01_01zone_material00
initForm
keyPressed

```
// ///////////////////////////
// E+ input

  // define a name for the zone (useful for plotting)
EnergyPlus.setName(zone, "zone");

// set E+ surface types (default is WALL)
EnergyPlus.setType(zone.faceEnd(), EnergyPlus.ROOF);
EnergyPlus.setType(zone.faceEnd(1), EnergyPlus.FLOOR);

// sets the boundary condition, i.e. indicate that the f
EnergyPlus.setBoundary(zone.faceEnd(1),EnergyPlus.GROUND

// set constructions (material layers) using predefined
for (int i=0; i<4; i++)
  EnergyPlus.setConstruction(zone.face(i),"Concrete Exte

EnergyPlus.setConstruction(zone.faceEnd(),"Medium Roof/C
EnergyPlus.setConstruction(zone.faceEnd(1),"Concrete Sla
```

```
Adding:
C:\Users\mark\Dropbox\sketchbook\5_examples_workshop\5_exa
mples_workshopCD2-12_4\EPlus01_01zone_material00\initForm.
pde
```

Figure 1: Example design and analysis produced by the students (Vinh Nhy Ly) for the final assignment, in which they were asked to design a parametric shading system adapting to different orientations (left). Example script within the Processing IDE, focused at EnergyPlus related functions (right).

through scripting (LaBelle et al., 2010). This library also provides an extensive set of functions that interface with EnergyPlus, allowing geometrical elements to be supplemented with simulation data to automatically produce EnergyPlus input files, launch simulation runs and analyze results (Nembrini et al., 2011). The script thus constitutes a complete description of the building model, including geometry and simulation parameters (Fig. 1, right).

A file is referred to in Processing as a 'sketch', reflecting the emphasis on a visual, design metaphor for the computational design process. This visual interface is used in the ANAR+ library as a means of viewing 3D geometry, adjusting user-defined parameters, running simulations and visualizing simulation results. Nonetheless, the sketch itself is a text file, and users have to become accustomed to defining geometry and parameters related to simulation using a purely text-based approach.

In order to record usage data, the ANAR+ library was modified to log the following details on a server each time a script is run within the Processing IDE:

- the user ID and time stamp,
- the name of the script being run,
- interactions with the GUI such as simulation runs or parameter modifications through sliders.
- the current state of the script's code.

Comparing the development of the script with BPS simulation run requests enables the reconstruction of the development process of each user's design process at a very fine-grained level.

Taking advantage of the text-based nature of the data recorded, source code version control software was used to store the usage history. With such a database it becomes possible in a second step to reconstruct the history of each user, especially the triggering of simulation runs. This approach allows us to easily identify any modifications to previously recorded scripts.

Teaching context

The context for our study is a year-long course in computational design which was offered in the 2011-2012 academic year as a required component of the MSc in ——— at the ——— School of Architecture. We provided basic instruction for the students to script using Processing. The target audience of the course were architecture master students enrolled in a sustainable design program — interested yet non-expert users who agreed to participate in the usage data recording. All students in the computational design course had a professional degree in architecture or a related design discipline, and all had been exposed to concepts of BPS in a required course on this topic.

The data described in this paper was collected during the second semester of the computational design course and the following summer, when some students used Processing/ANAR+ in their dissertations. During these two semesters students learned to use parametric scripting as a tool for generating architectural geometry and running EnergyPlus simulations as a means of testing the implications of their design decisions on thermal performance. By the beginning of the data collection period students were familiar with the basic principles of parametric scripting and had an introduction to the use of feedback from BPS tools in the architectural design process. All students gave their written approval to the data collection, which has been conducted and stored according to appropriate ethical procedures concerning research on human subjects.

The semester was organized as follows: an initial revision of parametric design principles ending with an

sign modifications and simulation runs. It was initially our expectation that sketches which combined a high frequency of simulation runs with a large number of code modifications and GUI interactions would be most likely to reveal links between simulation feedback and design development. We were also interested in understanding the number of unique users per sketch — copying of code by multiple users was taken as an indication of overall usefulness and thus as an indicator of the value of a given sketch for the users. The time span over which a given sketch remained in active use was also interesting to us as an indication of the continued utility of a given block of code through multiple different design contexts.

This visual data analysis proceeded in two different stages using the individual sketch as the unit of investigation: firstly considering the whole dataset and looking at the evolution of sketches over time (Fig. 2 and 3); and secondly looking at patterns in the use of sketches by individual users (Fig. 4 to 6). Finally, the analysis of a singled out user exploit version control software to track code re-use in new projects (Fig. 7).

Currently at an early stage in the analysis, focus has been put on patterns in the use of examples provided to the students as a basis for learning coding concepts, as well as on patterns of simulation usage in relation to script modifications.

The first step in our analysis focuses on visualizing the number of simulation runs in relation to the total activity recorded for a given script. Basing itself on the script's name, the occurrence of each script in time is related to the number of logging events, concentrating only on scripts that include BPS functions. These are plotted using the time difference between their first and last occurrence in the log against the number of log events concerning that specific script, displayed with logarithmic scales (Fig. 2). The visualization distinguishes between scripts initiated by the students and scripts provided by the module instructors in order to identify temporary patterns in the use of code over the course of the semester. The total number of simulation runs per script is also displayed. This graph demonstrates the fundamental difference between provided examples, which understandably have the greatest duration over time; and user-defined scripts which are more abundant and have a more even distribution in terms of time-span. Not surprisingly, scripts with the greatest duration also tend to have the highest level of activity in terms of simulations and total log events.

A similar view is provided in Fig. 3, which plots the number of unique users for a given sketch against the total number of simulation runs. Distinct usage patterns for user-initiated scripts and given examples are made clear: there is a split between scripts with larger numbers of users (examples) and scripts with 1, 2 or 3 users which are exclusively user-generated.

Our intention in producing such visualizations was to single out individual sketches that represent interesting approaches to parametric design in relation to simulation. However, as users tend to rename scripts to safeguard early variants as will be seen in the second step of the analysis, this visualization approach of focusing on the individual scripts as the unit of analysis falls short in capturing the actual design process taking place. As an example, sketches with the most simulation runs actually correspond to the rapid production of output for the final assignment presentation rather than illustrating the user's design process.

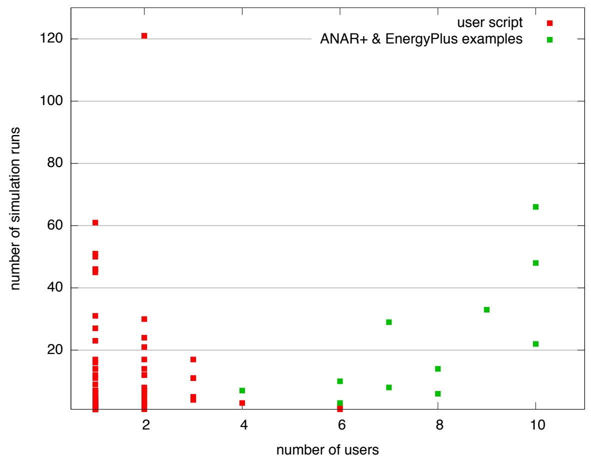

Figure 3: Simulation runs compared with number of users. A clear distinction is made between examples and user scripts. It is interesting that the greatest concentration of scripts with the highest number of simulation runs is in the category of sole-authored scripts. The single script displaying more than 120 simulation runs is an outlier bearing a generic name provided by the Processing IDE.

In a second stage of visualization we focused on the individual user as a unit of analysis. Concentrating on individual users it becomes possible to visualize differing patterns, both for the usage of examples as for the relationship between simulation runs and code modifications. Fig. 4 represents usage patterns of three different users, illustrating the difference between user-defined sketches, provided examples, examples including simulation functions, and simulation runs. In these graphs the unique ID assigned to each individual script is in the Y-axis, so horizontal lines of activity correspond to a particular script being run repeatedly by the user (scripts must be run to visualize the geometric relationships described in code). The concentric circles surrounding specific script runs indicate the number of lines changed compared to the previous run of that script.

Students tend to rely on examples to learn techniques early in the semester and come back to them when engaged in the design phase. Most examples involving simulation were introduced in a workshop in late March (shown as a vertical line on Fig. 4). Straight vertical stacks of activity indicate users working with multiple scripts open simultaneously or opening multiple scripts in rapid succession, while horizontal lines indicate activity within a single script over time.

For users 7 and 12 in Fig. 4, there appears to be a progressive increase over the semester in the number of lines changed, as one might expect as the student gains familiarity and confidence with coding. For other users (03 in Fig. 4) the writing of original code is more evenly distributed during the semester. The use of example scripts provided by the course instructors to illustrate key coding concepts was most intensive at the beginning of the semester. All three users went through a period of intensively using example scripts prior to the first assignment deadline, and all returned to the example scripts in the three weeks preceding the final assignment.

The 7-day period preceding the final assignment is focused on in Fig. 5. The return to provided examples can be seen across all users, with particular reliance on the examples that include the use of simulation; this return is particularly striking for users 03 and 07. Some choose to modify directly the example script, renaming the script in a second stage (note the code modification on the example in user 07 in Fig. 5). The running of example scripts parallels the development of the users' own code, and use of example scripts often corresponds with periods of frequent simulation runs using the user scripts. For each user there is one or two example scripts that they repeatedly return to during the process of developing their own scripts.

Finally, selected time lapse are closely scrutinized to illustrate differing patterns in simulation and variants usage (Fig. 6). In the development of their own scripts users 07 and 12 follow a mostly linear process, focusing on the same one or two scripts in the last hours before the assignment deadline. User 03 runs a number of different scripts during the last day before the deadline, most likely variants on the same script which have been given different names. Distinctive patterns of simulation usage can be observed as some users repeatedly check their design's consistency against simulation (users 03 and 07 in Fig. 6) while others alternate between periods of geometric definition and simulation runs (user 12).

Taking advantage of the data collected, the 24 hours preceding the final assignment are closely analyzed to unveil the relationship between code modifications and simulation runs. Fig. 7 schematically represents this analysis for user 03, corresponding to the period represented in Fig. 6 (top): each rectangle represents a unique version of each script while arrows between them indicate amounts of simulation being run and modifications to the scripts — categorized as modifications on geometry, on simulation definitions, and other changes to the script's code. Additional dashed arrows indicate on which previous script the new script is based. During this period of time 92 simulation runs were triggered by the user.

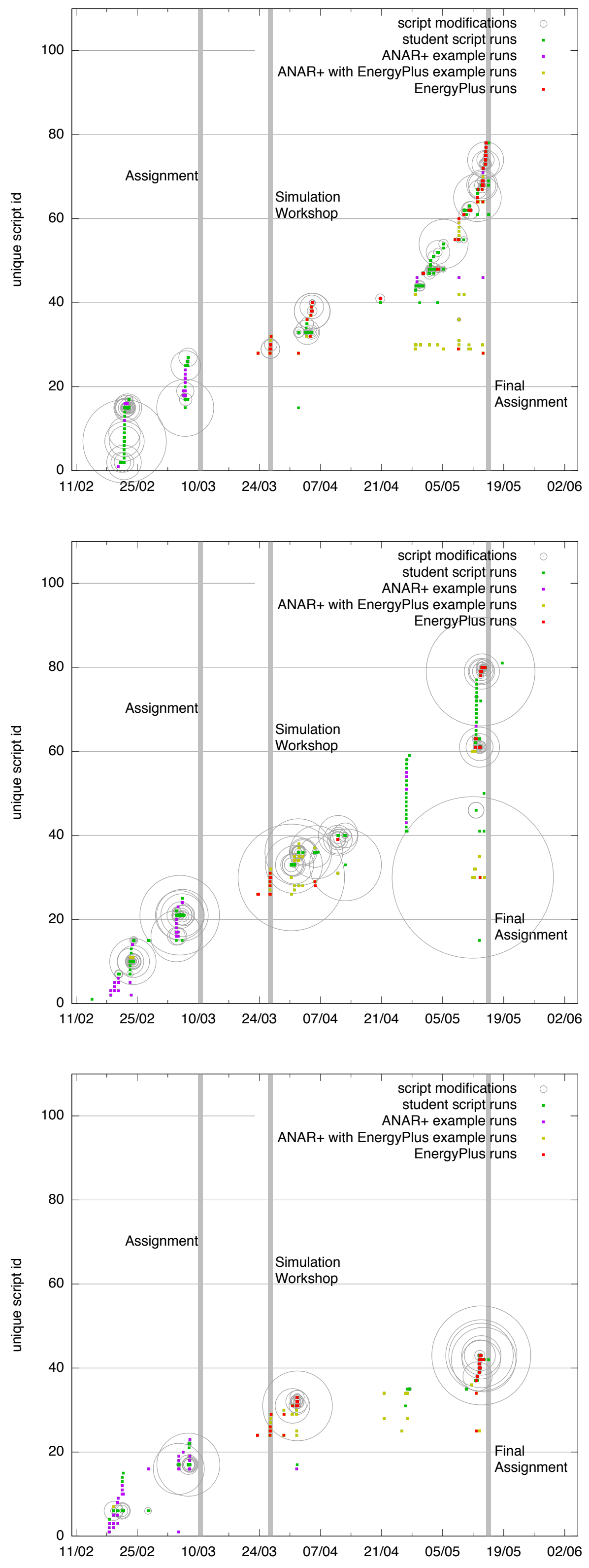

Figure 4: Time line of the Spring 2012 Workshop for users 03 (top), 07 (middle), *and* 12 (bottom) *over approx. 16 weeks. Differentiation is made between running examples scripts, examples scripts including simulation functions, user's own scripts and the triggering of simulation runs. Empty circles of variable sizes outline the amount of script modifications.*

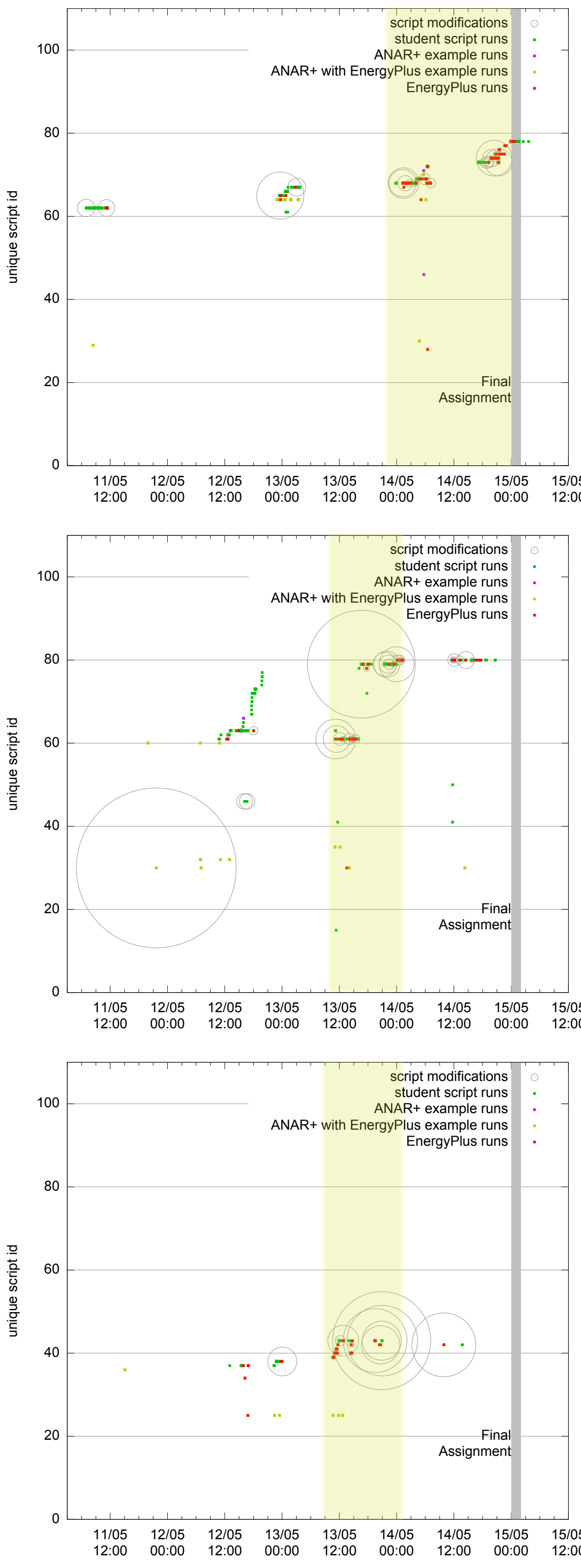

Figure 5: Time-line close-up for users 03 (top), 07 (middle), and 12 (bottom), preparing for final assignment over a period of 4 days.

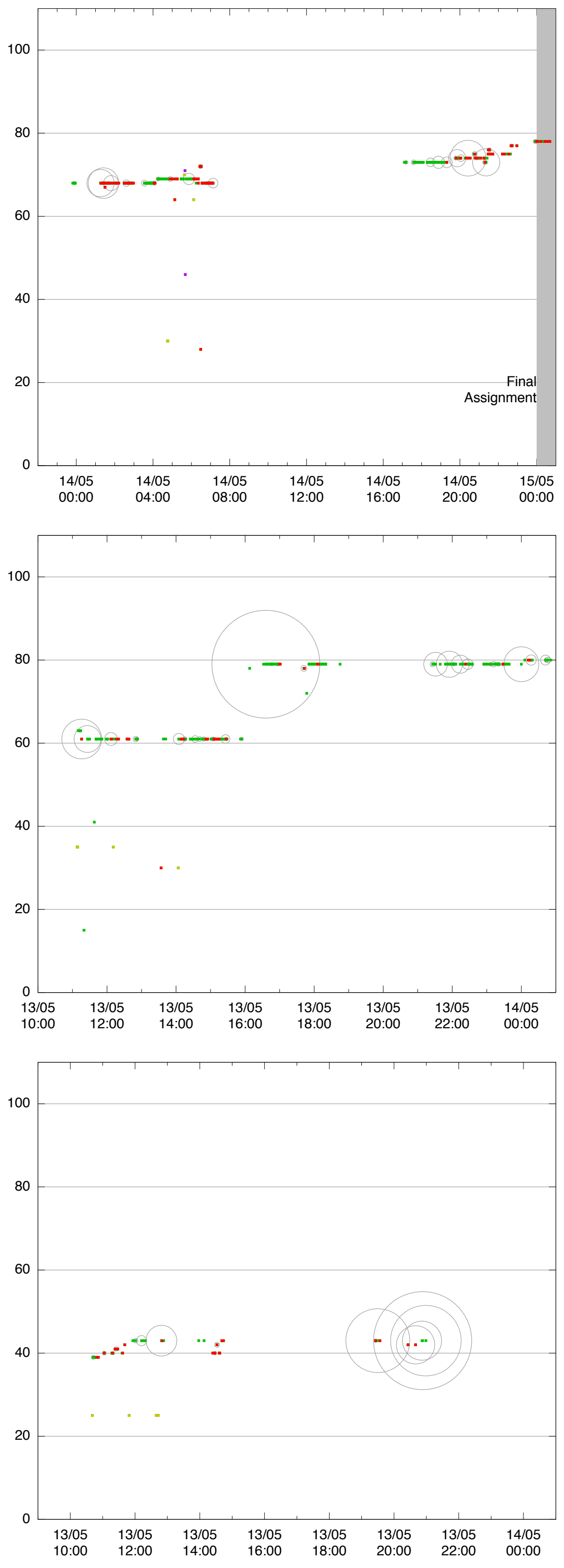

Figure 6: Further close-up on periods displaying specific activity, for users 03 (top), 07 (middle), and 12 (bottom) over a period of 16 hours.

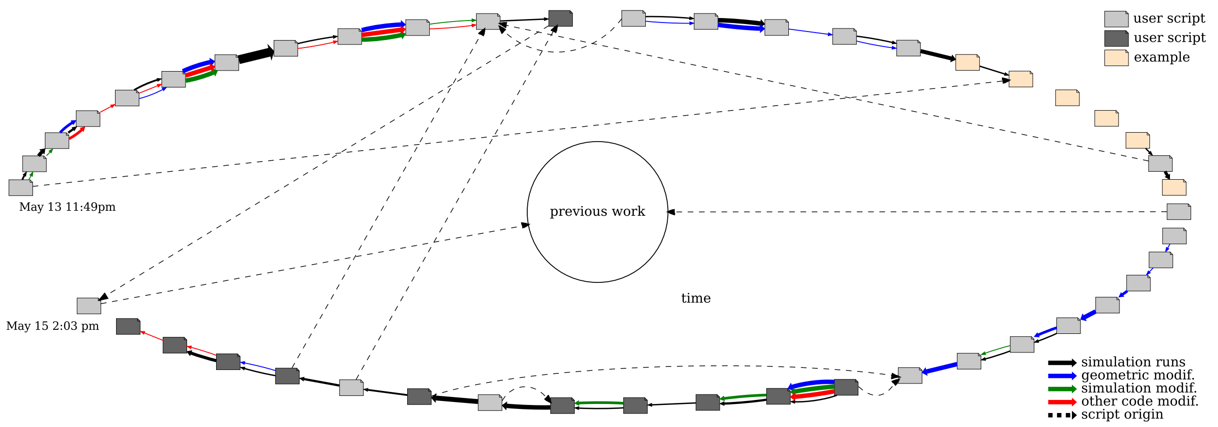

Figure 7: Detailed analysis of user 03*'s work during the 24 hours preceding final assignment, with each rectangle representing a unique version of a script. Several measures are depicted such as the amount of simulation runs between versions (black arrows), the amount of modifications between versions (blue, green, and red arrows). The dashed arrows indicate the script at the origin of a given script.*

Following the time, the dashed arrows make clear that the user does not follow a linear path. Instead, after developing a new script, he may fall back on the previous one, sometimes calling out a few examples, even starting again from older work.

Looking at the sequence of simulation runs, it is clear that they tend to happen in bursts rarely accompanied with significant modifications. Thorough work on the scripts tend to precede or follow intensive simulation requests. The lower right hand side of fig. 7 presents an interesting example in which geometry modifications are incrementally run for visual feedback and then tested through simulation, leading to further geometric refinement. Then the user engages in a new version with significant modifications which will be eventually thoroughly tested through numerous simulation runs.

DISCUSSION AND FURTHER WORK

One of the most important findings from the data is the importance of the provided examples in the process of designing through scripts. When engaged in the design phase, a majority of users periodically went back to examples, probably to understand/copy the code used to produce a certain functionality. Moreover, all contributions to the final assignment can be traced back to some provided example, sometimes even presenting quite a limited share of user-written code. This further confirms the importance of examples in the scripting approach. Code examples play a particularly important role in the computational design process as an embodiment of knowledge: both as know-how that elucidates the steps required to accomplish a complex task, and as a form of precedent or design example (Iordanova and Tidafi, 2007). This implies importance of carefully crafted examples that accompany the learning process, providing sound references as well as introducing advanced topics and allowing the student to read through and identify specific parts to be modified/worked upon.

An important weakness of the results presented is inherent in the data gathered, resulting from the relative simplicity of the task asked to the students. The final assignment brief posed to the students focused on shading, orientation and window/wall ratio; the interplay with thermal mass, insulation, internal gains, etc was not considered for the most part by the students. Further work will consider more systemic problems, possibly using the presented method to the competition settings between participants as described in (Reinhart et al., 2012). Such an improvement would further allow us to compare performance achievements in relation with simulation usage, as current performance results are not convincing enough. The fact that the user base is a captive audience is another weakness: in real design practice the question is open to use or not simulation, whereas students had to use it within the course.

The strength of the visualizations presented in this paper has been in providing a graphic representation of differing approaches to parametric design with simulation feedback. Using the visualizations of activity for a particular user, we have been able to recognize several distinct patterns of interaction between iterative cycles of parametric modification and simulation. In a future stage of the research, we will continue to examine this dataset and data collected in other classes and workshops with the goal of achieving through the data analysis a detailed understanding of how users learn from and respond to simulation results in a parametric design process.

If we are to extract more fine-grained information about the design process, we will have to dedicate more attention to the scripts themselves and the story they tell about the design process. The analysis presented here has mainly focused on time-based quantitative information, acquired through the help of software computational tools. Time has been missing to explore in depth the individual scripts themselves to uncover semantic information in relationship with the use of BPS. Some initial work in this direction has unveiled very rich information which requires a formalized approach. Further work is planned which will employ the Function-Behaviour-Structure (FBS) ontology on the recorded scripts, monitoring the process of creativity using a comparative technique (Gero and Kannengiesser, 2004; Pourmohamadi et al., 2011). The final aim will be to define automatic schemes to extract specific usage affordances on the whole dataset and produce dedicated measures inherent to the FBS ontology, such as Markov transition values or sliding window entropy which have been proposed as an estimate of design creativity. Such analysis would actually provide a measure of the appropriation of the tools by the designers according to their preliminary skill, and express to which extent the BPS tools actually influence the design process.

CONCLUSION

This paper documents an ongoing study of simulation in the early stages of the scripting-based parametric design process. The goal of this study has been to analyze in detail the design process followed by students in order to assess the potential of parametric techniques as a means for designers to test, adjust and better understand the results received from simulation engines. Through an initial analysis of an unprecedented usage dataset, the work represents to our knowledge the first study of its kind on a design scripting framework in conjunction with BPS. Despite needing further work, the research already reveals a number of distinct design processes in relation with BPS usage. By outlining pitfalls and advantages of using scripting in this context, it is our expectation that this method will allow us to formulate specific guidelines required to integrate these tools in early stage design. Further work will concentrate on formulating more demanding tasks to students to induce greater need of BPS results interpretation and feedback in the design process. The approach presented here is believed to be of significant importance for the general aim to provide designers with explorative tools to consider sustainable construction in a systematic and informed manner.

ACKNOWLEDGEMENT

The authors wish to thank ——— and the University of ——— (UK) for providing their knowledge and infrastructure to record the students' data. ——— is supported by the ——— grants ——— and ———
.

REFERENCES

Attia, S., Beltran, L., Herde, A. D., and Hensen, J. 2009. "architect friendly": A comparison of ten different building performance simulation tools. In *Building Simulation 2009, 11th IBPSA Conf.*, pages 204–211, Glasgow, Scotland.

Attia, S. G. and Herde, A. D. 2011. Early design simulation tools for net zero energy buildings: A comparison of ten tools. In *Proceedings of Building Simulation 2011: 12th IBPSA Conf.*, Sydney.

Burry, M. 2011. *Scripting cultures: Architectural design and programming*. John Wiley and Sons, West Sussex, UK.

Crawley, D. B., Hand, J. W., Kummert, M., and Griffith, B. T. 2008. Contrasting the capabilities of building energy performance simulation programs. *Building and Environment*, 43:661–673.

Gero, J. and Kannengiesser, U. 2004. The situated function-behavior-structure framework. *Design Studies*, 25(4):373–391.

Hopfe, C. J., Struck, C., Harputlugil, G. U., Hensen, J., and Wilde, P. D. 2005. Exploration of the use of building performance simulation for conceptual design. In *IBPSA NVL symposium*.

Ibarra, D. I. and Reinhart, C. F. 2009. Daylight factor simulations — how close do simulation beginners "really" get? In *Building Simulation 2009, 11th International IBPSA Conference*, Glasgow, Scotland.

Iordanova, I. and Tidafi, T. 2007. Referents modeling for the architectural design studio: Cognitive bases. In *Proceedings of EuropIA-12*, pages 171–184.

Jakubiec, J. A. and Reinhart, C. F. 2011. Diva 2.0: Integrating daylight and thermal simulations using rhinoceros 3d, daysim and energyplus. In *Proceedings of Building Simulation 2011, 12th IBPSA Conf.*

LaBelle, G., Nembrini, J., and Huang, J. 2010. Geometric programming framework: Anar+ geometry library for processing. In *Proceedings of eCAADe 2010, Future Cities*, pages 403–410.

Nembrini, J., Samberger, S., Sternitzke, A., and LaBelle, G. 2011. The potential of scripting interfaces for form and performance systemic co-design. In *Proceedings of Design Modelling Symposium*, Berlin. Springer.

Oxman, R. 2008. Performance-based design: Current practices and research issues. *Int. Jour. of Architectural Computing*, 6(1):1–17.

Pourmohamadi, M., Gero, J., and Saunders, R. 2011. Cad software as customisation tools. In Herr, C. M., Gu, N., Roudavsky, S., and Schnabel, M. A., editors, *Proceedings of the 16th Int. Conf. on CAAD Research in Asia (CAADRIA) 2011.*

Reinhart, C. F., Dogan, T., Ibarra, D., and Samuelson, H. W. 2012. Learning by playing — teaching energy simulation as a game. *Journal of Building Performance Simulation*, 5(6):359–368.