Educational and Research Directions in Design Computing and Virtual Architecture

John S. Gero, Mary Lou Maher and Rabee M. Reffat

Key Centre of Design Computing and Cognition

University of Sydney, Australia

Email {john, mary, rabee@arch.usyd.edu.au}

Abstract

This paper describes research and educational paradigms of Design Computing and Virtual Architecture. The paper provides examples of research projects and educational programs at the Key Centre of Design Computing and Cognition, University of Sydney, along with preliminary results obtained through different paradigms. These paradigms include cognitive and computational models.

1 Introduction

Design Computing (DC) and Virtual Architecture (VA) have generally been looked at as a subset of computer applications that assist designers in modelling, presenting and analysing designs. Currently, most Computer-Aided Architectural Design (CAAD) applications are used at the later stages of the design process after the crucial decisions of creating design artefacts have been considered. DC and VA have generally been driven by technological developments in which they relied on software developers to implement and market software with relevant features and utilities that support some aspects of design activities. In this paper, we consider DC and VA as research areas in which the results of the research lead to more than additional computer programs. Such results would lead to a better understanding of designing and computer support for designing. The foci of these research are: (a) developing theories, models and methods of designing; (b) using these theories, models and methods as the basis for the development of computational tools; and (c) using these theories, models and methods as the basis for teaching. The first set of goals has more to do with design research rather than strictly design computing research. In order to achieve the first set of goals, it is sometimes useful to consider computational models of design as a way of simulating design processes. However, human designers can also provide the basis for developing theories, models, and methods of designing. The second set of goals looks at the implications of particular theories, models, and methods of designing when considering computer support or automation of specific design tasks. This set of goals has a more direct correlation with the majority of design computing research currently taking place at universities. The third set of goals brings this understanding of design processes to bear on how we teach design.

The research in Virtual Architecture involves the development of representational models

as a metaphor for virtual worlds and 3D virtual environments. Various digital media, agent technologies, and interactivity between people are explored in these representational models. Designing virtual places is viewed as developing computer-mediated dynamic worlds that create a sense of place.

In this paper we approach teaching Design Computing in a coherent and pedagogical process that is more than teaching the techniques of using certain CAAD software applications.

2 Research Paradigms in Design Computing

The paradigms that we found useful and distinctive in pursuing research in Design Computing include cognitive and computational models. Cognitive models are empirically-based research that involves the development of experimental studies of designers while designing. Computational models are axiom-based or conjecture-based research. Axiom-based research involves the identification of a set of axioms and their consequences to derive logic-based computational models of designing. Conjecture-based research involves an analogy between a cognitive or computational process that leads to computational models specific to designing.

This paper describes the characteristics of each of the three paradigms and gives examples of research projects at the Key Centre of Design Computing and Cognition at the University of Sydney that illustrate the approach and preliminary results obtained through the different paradigms.

2.1 Empirically-Based Design Computing Research

Empirically-based research uses the experimental paradigm in which experiments are set up and then data is collected and analysed to produce a set of results. These results are then used as the basis of either the development of a hypothesis or the confirmation of a hypothesis about designing. The experiments are typically developed to provide evidence for a particular theory or cognitive model of designing. Typical approaches to empirically-based design computing research are: direct observation of the results of designing; surveys of designers' perceptions; and protocol studies of individual and collaborating designers designing. New protocol analysis methods have been developed and are being applied to produce novel results concerning the behaviour of designers as they are designing which has significance for the development of computational tools for designers.

2.1.1 What designers do when they are designing?

Designers were asked to carry out a specific design task and the "think aloud" as a protocol analysis method was employed. Each designer was videotaped and a rich coding scheme was developed based on both design theory and the need to accommodate the data in the transcription. The development of the coding scheme is a crucial aspect of the protocol analysis method. The coding scheme developed here used five generic categories. The advantage of the use of categories is that they allow for an additional confirmation phase in the analysis since they exhibit interdependence. The five categories developed were (Gero and McNeill, 1998): problem domain - abstraction level; function-behaviour-structure;

analysis and evaluation; synthesis micro-strategies; and design macro-strategies.

Protocol analysis results

At a gross level a designer's time can be spent either on postulating solutions, called structure, or in reasoning about the function and behaviour of possible or postulated designs. Figure 1 shows a typical distribution of the time spent between these two large classes of activities by a designer. It is interesting to note that it is almost twenty minutes into the session, for this design, before any structure is proposed.

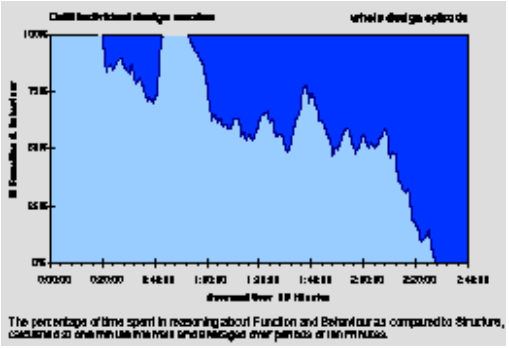


Figure 1. Typical plot of distribution of time spent on function and behaviour (light), as against structure (dark), for an experienced designer (Gero and McNeill, 1998).

Considerable detail about various aspects of designers' behaviour can be determined using the protocol analysis method. A surprising finding in this experiment, from the analysis of the spectrum of design event lengths across a typical design session, is the very short duration of each design event. Without experiments with human designers such information would not become available.

2.1.2 Insights of designers' actions

Suwa et al (2000) looked into the cognitive processes of a practicing architect using the technique of protocol analysis. The protocols of his design session were collected as a retrospective report after the session. The design session, which lasted for 45 minutes, was to work on the conceptual design of a museum on a given site in a natural environment in the suburb of a large city. The architect was encouraged to draw sketches on tracing paper. His sketching activities were videotaped. In the report session, he talked about what he had been thinking of for each stroke of his pencil during the design session, while watching the videotape.

In cognitive science, there has been a prevailing view that human cognition is a situated act and that physical performances, representing the world, perceiving and conceiving are dynamically coupled and they, as a whole, form a coordinated cognitive activity (Clancey, 1997). The findings in this experiment provided empirical evidence for this statement in many ways, and therefore suggest that designing is a situated act. First, the architect invented design issues or requirements not just by the use of explicit knowledge, but also by constructing justifications or reasons for them on the fly during the process. The construction of those justifications or reasons was dynamic in the sense that the architect

did so through unexpected discoveries of unintended visuo-spatial features of the developing solution-space, i.e. design sketches. Making unexpected discoveries is seen as the act of re-representing the visual field in the sketch. This way, the emergence of conceptual ideas, i.e. design requirements in this case, is situated in the acts of representing and perceiving. Unexpected discoveries of unintended visuo-spatial features are entirely dependent on what the designer has so far drawn in sketches, and thus on what kinds of configuration of drawn elements the designer sees in front of him or her at the moment. The emergence of a conceptual idea enabled the architect to see his own sketches from a new point of view, and thus encouraged the generation of a new perception. This indicates that a designer's perception is entirely dependent on, or coupled with, his conception.

2.1.3 Cognition-based CAAD

Protocol analysis is utilised to examine the design processes in order to provide information for a cognition-based CAAD system. Retrospective protocols has recently been used in several studies to explore human design activities since they minimise the interference caused to the designers. This study (Tang and Gero, 2001) follows the same method in which participating designers first design for the designated brief. An expert and a novice were videotaped then retrospectively they reported the design process with the aid of the videotapes of their designing in the first phase.

In the empirical data, shown in *Table* 1, the expert's encoded protocol consists of 338 segments with an average time span of each segment of 8 seconds, whereas the novice's encoded protocol consists of 145 segments with an average time span of 20 seconds. The experimental duration was 45 and 48 minutes respectively.

Table 1. Number of segments and average time span of the novice and the expert

	Number of segments	Average time span (seconds)
Novice	145	20
Expert	338	8

The result indicates that the shift of focus from one topic to another was on average every 8 seconds for the expert and 20 seconds for the novice. The speed of shifts is much faster than what was expected. In terms of using a CAAD system, the time this expert took to shift his focus was of the order of that for a user to pull down a menu, select the function, and input parameters. The surprisingly fast speed of change of topic during conceptual designing provides a basis for why designers during conceptual designing still prefer using pen and paper even when expensive, powerful, and cutting-edge CAAD systems are available for their use. It is simply because sketching skills using pen and paper allow them come up to the speed of thought, follow their ideas, and be creative. The speed here is not relevant to computational power nowadays because even the latest CAAD system cannot efficiently support this design phase. The interaction between designers and machines is not sufficiently intuitive and simple enough to follow the train of thought so that the use of a CAAD system does not match the development speed of thought and ideas.

2.2 Axiom-Based Design Computing Research

Axiom-based research produces computational models of design through the identification

of a set of axioms and the logical consequences of the axioms. This approach to design computing research involves: specifying relevant axioms; deriving logical consequences of the axioms; mapping the axioms and their consequences onto a particular domain to derive new results. For example, an axiomatic logic-based shape representation allows for the uniform representation of shapes with or without curved boundaries, the consequences of which are representations of complex shapes that can be manipulated with logical implications (Damski and Gero, 1996). Consider the universe of discourse as the space defined in Figure 2. The axiom is that the space can be divided into two complementary spaces.

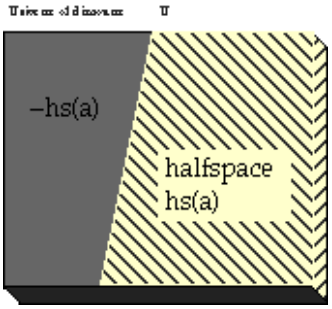


Figure 2. A space divided into two halfspaces, labelled hs(a) and -hs(a).

The following can be defined or inferred from the axiom: a predicate *hs(a)* is defined for the halfspace *a* and *-hs(a)* for the halfspace *a'*, *hs(a)* is defined as True and *-hs(a)* as False

a volume *V* is $hs(a1) \wedge hs(a2) \wedge ..... \wedge hs(an)$

a shape S is *V1* / *V2* / *V3* / ..... / *Vm*.

2.3 Analogy-Based Design Computing Research

The development of theories, models and methods of designing often relies on identifying an analogy with other processes. This research paradigm starts with a relevant computational process or cognitive model of design and develops a specific computational model of design. Some examples of computational models based on an analogy with cognitive models of design include: case-based design (design based on precedents; representation of cases including multimedia representations); design prototypes (knowledge chunking); graphical emergence (emergence of shapes, objects, semantics and style from drawings); design by analogy (between domain analogies in particular); and qualitative reasoning in design (qualitative representation and reasoning about shapes and spaces). The development of computational models of designing need not rely entirely on cognitive models of designers, there is the potential to identify an analogy with other computational processes and apply them to a design domain. This type of research borrows heavily from computing fields such as AI and Operations Research to produce specific computational models of design; for example: evolutionary systems (genetic engineering and co-evolution); and neural networks (emergence models).

2.3.1 Case-based design

Case-based reasoning provides design support by reminding designers of previous experiences that can help with new situations. Designers learn to design by experiencing design situations. This is reflected in the way we teach students to be designers: engineers are taught to have analytical capabilities and then learn to design in professional practice, architects are taught through exposure to a range of design experiences in the studio. We learn to analyse through the use of formal methods, but creating a new design requires previous experience, or at least, exposure to another's design experiences. As a cognitive model of design, case-based reasoning provides the basis for a computational model of design. Case-based reasoning as a support environment for conceptual design is attractive for two reasons: the knowledge is represented as design cases that can be proprietary and/or familiar to the designer; and the knowledge as case memory can be maintained and updated automatically with the use of the system. The application of case-based reasoning to structural design, eg CASECAD and CADSYN (Maher et al, 1995), has shown that the development of these case-based reasoning systems has to take into consideration a formal representation of design cases and the knowledge

2.3.2 Shape emergence

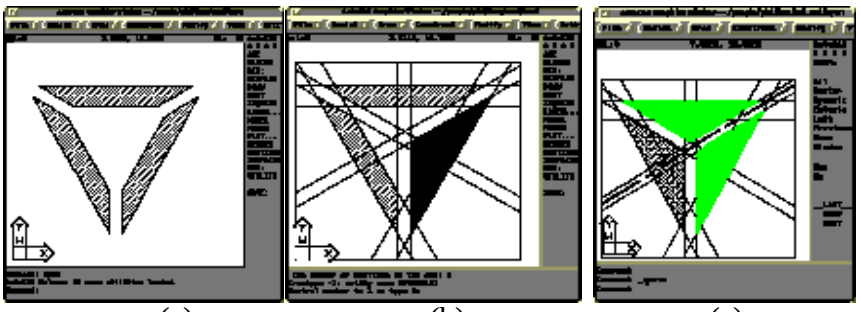
Emergence is the process of making properties, which were previously only implicit in a representation, explicit. Figure 3 clearly demonstrates the phenomenon. If the right-hand figure is drawn using a CAD system, its representation will be that of six objects located in geometrical space. However, for humans the dominant features are the central star and triangles. None of the features seen by the human observer can be "seen", ie, are represented by the CAD system.

Figure 3. An object and a composite object, made of six copies of the single object, which exhibits strongly emergent shapes.

From the work of the Gestalt psychologists and more recently that of the cognitive psychologists, it is possible to construct computational models of shape emergence based on concepts drawn from their research. Humans appear to distinguish foreground from background in their reading of shapes. In order to emerge shapes which were not previously represented a process which manipulates the foreground and background can be constructed. What is done is to take the primary or originally represented shape and "unstructure" it so that it now becomes part of the background, producing an image composed of unstructured shapes only. A structuring process is then passed over this background to emerge foregrounds which may include both the primary shape and newly represented shapes. Gero and Yan (1993) have developed such a process based on a new representation, infinite maximal lines, along with a structuring process.

The concepts behind shape emergence can be extended to emerge shape semantics, where the shape semantics are derived from visual patterns of shapes. Since these patterns were not originally represented they are emergent when there is a computational process which can find and represent them. From seeing drawings, various visual patterns are perceived by the human viewers. Designers can find different visual patterns from what was intended to be drawn. The newly discovered visual patterns may play a crucial role in developing further ideas in the same design if the designer is willing to adapt the visual pattern which was not there at the moment of drawing. Regardless of adaptability, visual patterns from shapes are defined as *shape semantics* when the patterns match the criteria for predefined labels, such as visual symmetry, visual rhythm, visual movement and visual balance.

Gero and Jun (1998) have developed a computational model of shape semantic emergence which is based on three processes: object correspondence, grouping, and shape semantics emergence. In order for shape semantics to exist there needs to be some form of structured regularity in the overall image. Object correspondence is the process which locates regularity of shape repetitions. Grouping locates regularity of groups of shapes, whilst the final process is hypothesis-driven and attempts to find known regularities amongst the groups of shapes. In Figure 4 (a), the initially drawn image is shown. Figure 4(b) shows that an emergent shape has been found. Figure 4(c) shows that the shape semantic reflectional symmetry has been found.



(a) (b) (c)

Figure 4. (a) The primary shape as input in AutoCAD; (b) Discovering emergent shapes; (c) discovering and using reflectional symmetry based on the emergent shapes (Jun and Gero, 1997).

2.3.3 Situated learning in design: application to architectural shape semantics

Situated learning is based on the notion that knowledge is more useful when it is learned in relation to its immediate and active context, ie its situation, and less useful when it is learned out of context. The usefulness of design knowledge is in its operational significance based upon where it was used and applied. A computational system of **S**ituated **L**earning **i**n **De**sign (SLiDe) (Reffat and Gero, 2000a) was developed to elucidate how design knowledge is learned in relation to its situation, how design situations are constructed and altered over time in response to changes taking place in the design environment. SLiDe is implemented within the domain of architectural shapes in the form of floor plans to capture the situatedness of shape semantics. SLiDe utilises an incremental learning clustering mechanism that makes it capable of constructing various situational categories and modifying them over time.

SLiDe is a system that locates design knowledge in relation to its situation. It modifies its behaviour as its situation from the design environment changes. SLiDe is an active system that responds to dynamic changes in its environment. It selects appropriate actions as a response to its immediate situation through recognising various contexts to which it is potentially situated. SLiDe structures its encountered situations and classifies them into situational categories in a hierarchical manner.

Developing a computational model of situated learning in design to produce these kinds of situational categories provides opportunities to assist designers during the conceptual design process. One way is to integrate SLiDe with current CAD systems such as AutoCAD to make it easier for designers to conceptualise and explore their designs beyond the mere drafting of them. SLiDe helps to explore shapes of designed elements drawn in AutoCAD and allows the designers to have varieties of representations of what they have designed that may lead them to different moves than are they originally contemplated as shown in Figure 5 (Reffat and Gero, 2000b).

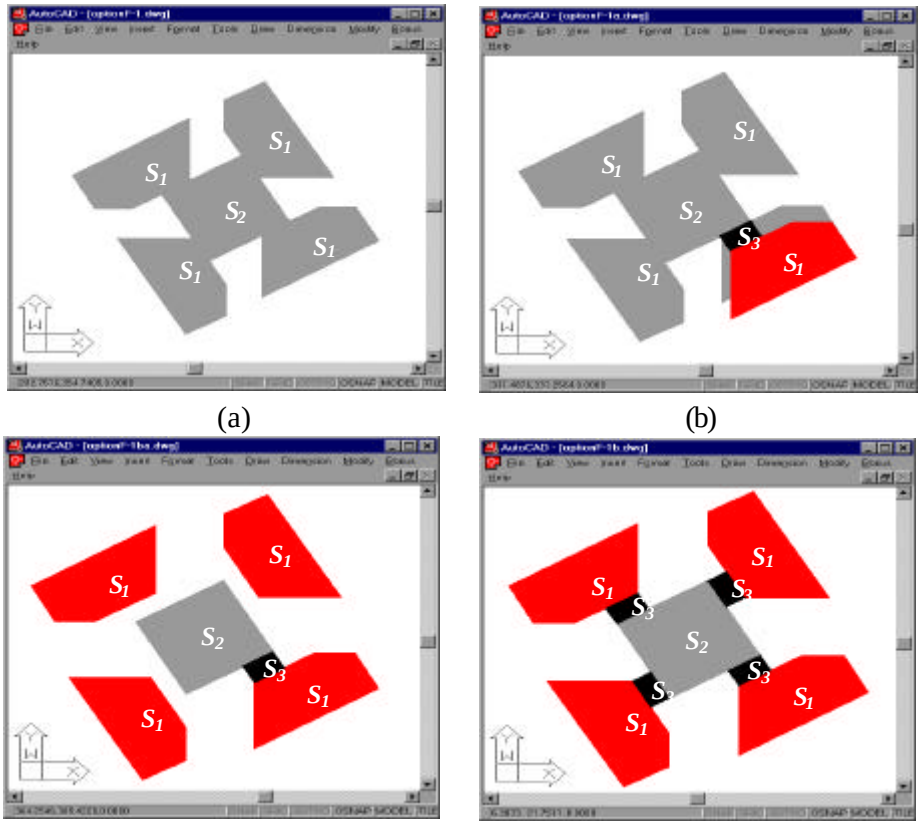


Figure 5. (a) A new move that a designer selected from the developed representations to pursue further in designing, (b) a new space added by the designer at a later stage, (c) SLiDe-CAAD could help in maintaining the integrity of cyclic rotation via preserving its necessary conditions, and (d) SLiDe-CAAD could help in maintaining the situatedness of cyclic rotation via preserving its applicability conditions, eg. adjacency and centrality (Reffat and Gero, 2000b).

These different representations of the same design help to arouse the designers' attention to

potentially hidden visual features of their design elements. This can be called, enhancing the perceptual interaction with design elements. Further, SLiDe can draw designers' attention to a set of shape semantics available in their current designs by highlighting a particular set of design elements that reflect those semantics that the designers may indicate which of these semantics attracted them. SLiDe, having stored the designer's interest as the focus can dynamically change the association between design elements during the design process by maintaining the situation of the designer's focus. So, whenever the designer made changes in the design after indicating the semantics of interest, SLiDe can change all the interrelated design elements to maintain the focus by maintaining the relationships that define the situation of that focus. Using SLiDe to provide such features in current CAD systems can potentially help to support designers in designing as well as drafting and at the same time have the potential to change the nature of current CAD systems from passive systems to active support design systems.

2.3.4 Curious design agents for novelty and creativity

Computational models of curiosity provide general-purpose, knowledge-lean heuristics to guide the search for potentially interesting, and possibly even creative, designs. A curious design agent is an agent that uses the search for novel designs to guide its design actions. Saunders and Gero (2001) developed computational model of curiosity based on a process called novelty detection. Determining interestingness depends upon the knowledge of the agent and their computational abilities; things are boring if either too much or too little is known about them. Hence situations that are similar-yet-different to previously experienced situations are the most interesting and this is what we mean when we say that something is novel. A novel situation is one that is similar enough to previous experiences to be recognised as a member of a class but different enough from the other members of that class to require significant learning. It is a relatively straightforward to develop a computational model of interest based on this definition of novelty. A very simple model of interest used in the following experiments maintains an average of the novelty detected over a fixed window of the ten most-recent situations. A boredom threshold is used to determine when the interest in the current area of a design space is low enough to warrant a change in the design process, e.g. a switch from problem solving to problem finding.

Architects increasingly face the problem of "information overload" as they try to explore complex design spaces for innovative solutions. Although generative design tools relieve some of the burden of designing, they can make the problem of information overload worse as designers attempt to understand the significance of the designs produced. Technologies similar to curious design agents may play an important role in future CAAD systems by reducing the number of designs presented to an architect to a subset of those that are judged to be potentially interesting. Providing design agents with motivations that reward the discovery of interesting designs more closely matches the motivations behind human exploration of design spaces. The application of curious design agents may allow future CAAD systems to provide more natural and rewarding collaborative partnerships between designer and machine.

3. Virtual Architecture and learning environments

Virtual architecture (Maher et al, 2000; Maher, Simoff, and Cicognani, 1999) is an

electronic representation of architectural design. The phenomenon of virtual architecture can have two purposes: a simulation of physical architecture or a functional virtual place. The simulation of physical architecture is the most common purpose of virtual architecture and is increasingly being used to visualise, understand, and present architectural designs. The second purpose of virtual architecture involves the design and creation of virtual places in terms of its functional organization and electronic representation. Architects design buildings to provide places for people to live, work, play, and learn. Such places are embodied as buildings with internal spaces called rooms, halls, theatres, etc. An emerging concept for designed virtual places is to provide an electronic location for people to socialise, work, and learn. The metaphor of buildings and rooms can be revisited and used in virtual places, suggesting the potential for virtual places to be designed by architects and then constructed by programmers.

From the early Dungeons and Dragons, a text-based virtual world, to Active Worlds (http://www.activeworlds.com), a 3-D immersive collaborative modelling world, we witness a gradual transition from textually described online environments through to virtual places that are described in 3D geometry, sounds and textures. Various design approaches have been developed to provide virtual world designers with a set of design principles and parameters in order for them to effectively design an online environment. Among them, text and graphical approaches (Cicognani and Maher, 1998) have been identified although possibilities also exist for other approaches. Our experience in designing virtual worlds reveals a process whereby precedents studies are carried out by reviewing the current state of virtual architecture followed by a formulation of a design brief (Maher et al, 1999). This indicates the importance of both, the product and process of architectural design in designing virtual worlds.

We designed a virtual conference room for use during the DCNet99 conference (http://www.arch.usyd.edu.au/kcdc/conferences). The conference room is part of our Virtual Campus (http://www.arch.usyd.edu.au:7778), and is used as a 3D representation of a place with slide projectors, shared whiteboards, and a chat-like talking capability. The directions for the design of virtual architecture include:

- The establishment of principles for the design of virtual architecture.
- The development of a representation framework for virtual architecture
- Experimentation with the use of virtual architecture.
- The development of a set of proactive objects that fulfill the needs in virtual architecture that are not possible in physical architecture.
- Consideration of the implementation of efficient virtual architecture.
- Development of 3D modeller and navigation tools that support the principles of virtual architecture.

4. Educational program in Design Computing

The educational approach of Design Computing is focused on the use and development of computational models of design processes and digital media to assist and/or automate various aspects of the design process with the goal of producing higher quality and new

design forms. Design Computing also provides a basis for studying formal methods of designing and their computational support. The future of design computing includes the design of cyberspace as an environment for professional collaboration, bringing the application of design computing from the design of physical objects to the design of virtual places.

Here again, the focus is not entirely on computer applications for design, but on the use of computational models and/or cognitive models of design to inform design teaching. The Key Centre of Design Computing at the University of Sydney carries out teaching and research in the area of design computing. There are approximately 300 undergraduate architecture students, 90 graduate design computing/digital media students, 15-20 doctoral students, and 10 academic and research staff at the Key Centre. The framework for design computing research presented here is based on research that has taken place at the Key Centre over the last 30 years.

A new undergraduate degree in Design Computing has commenced this year at the Faculty of Architecture, University of Sydney. The philosophy of this degree is to bring together three core concepts in design computing, united by the keyword "digital", allowing a student to specialise in one while being knowledgeable about the other two. These core concepts are: developing environments for designing digitally; designing digitally and interacting with designs digitally.

Developing environments for designing digitally involves a conceptual and practical understanding of current digital technology for design and can lead to the development of new methods and techniques for designing, including languages of designing. Designing digitally requires knowledge of the various ways in which designs can be represented and generated. Interacting with designs digitally is a new area that involves knowledge of computer-mediated collaboration and how designers interact with and via different digital media. The concept of virtual architecture as either a simulation of the physical world or as a functional virtual world, brings these three core concepts together.

Graduates of the Bachelor of Design Computing will become the new change agents in society. Specifically, graduates will be: specialists that support design computing activities; designers that support their own design computing activities; designers of computing environments and virtual architecture; teachers and educators in design computing, or researchers in design computing.

5. Directions for Design Computing Research

This paper has described a framework within which design computing research is carried out. A number of research projects from the Key Centre of Design Computing and Cognition, University of Sydney, have been presented as vehicles for each of these paradigms. Each of the projects uses one of the paradigms listed. The conduct of research for each of the projects is different and in some cases quite different. Empirically-based design computing research looks like experimental cognitive science research. Axiom-based design computing research looks like mathematical/logic research. Conjecture-based design computing research looks like theoretical engineering research. Thus, design computing research spans a range of research paradigms. What both the projects and the framework of paradigms imply is that design computing research has now reached a level

of maturity that allows it to operate as a sub-discipline of design science rather than as simply a means of producing software packages. In this it contributes directly to the three goals enunciated in the Introduction. It is one of the primary means of developing theories, models and methods of designing as a process. It uses these theories, models and methods of designing as a process as a basis for the development of design tools, and is beginning to use the theories, models and methods as a basis for teaching (Gero and Maher, 1997).

What directions are open for design computing research? As empirically-based research produces more results, we should have a greater understanding or how human designers design. Such knowledge will have implications for both how information technology can be interfaced with human designers and, perhaps more importantly, provide new conjectures for design computing research to explore in order to provide the foundation for more useful tools for designers. Similarly, as the other approaches yield insights into designing they may provide the foundation for novel tools.

References

Cicognani, A. and Maher, M.L. 1998. Two approaches to designing virtual worlds, Proceedings of Design Computing on the Net 98, *International Journal of Design Computing*, http://www.arch.usyd.edu.au/kcdc/journal Vol 1.

Clancey, W.J. 1997. Situated Cognition: On Human Knowledge and Computer Representations, Cambridge University Press, Cambridge

Damski, J. C. and Gero, J. S. (1996) A logic-based framework for shape representation, *Computer-Aided Design* **28**(3):169-181.

Gero, J. S. and Maher, M. L. (1997) A framework for research in design computing, *in* B. Martens, H. Linzer and A. Voigt (eds), *ECAADE'97*, Osterreichischer Kunst und Kulturverlag, Vienna (CD-ROM), Topic 1, paper 8.

Gero, J. S. and McNeill, T. (1998). An approach to the analysis of design protocols, *Design Studies* **19**(1): 21-61.

Gero, J. S. and Yan, M. (1993). Discovering emergent shapes using a data-driven symbolic model, *in* U. Flemming and S. Van Wyk (eds), *CAAD Futures'93*, North-Holland, Amsterdam, pp. 3-17.

Jun, H. and Gero, J. S. (1997). Representation, re-representation and emergence in collaborative computer-aided design, *in* Maher, M. L., Gero, J. S. and Sudweeks, F. (eds), *Preprints Formal Aspects of Collaborative Computer-Aided Design*, Key Centre of Design Computing, University of Sydney, Sydney, pp.303-319.

Jun, H. and Gero, J. S. (1998). Emergence of shape semantics of architectural shapes, *Environment and Planning B: Planning and Design* **25**(4): 577-600.

Maher, M.L., Balachandran, B. and Zhang, D.M., (1995). *Case-Based Reasoning in Design*, Lawrence Erlbaum, Hillsdale, New Jersey.

Maher, M.L., Simoff, S., Cicognani, A. 1999. *Understanding Virtual Design Studios*, Springer-Verlag, London.

Maher, M.L., Simoff, S., Gu, N. and Lau, K.H. (1999). Two approaches to the virtual design studio, Proceedings of Design Computing on the Net 99, *International Journal of Design Computing*, http://www.arch.usyd.edu.au/kcdc/journal Vol 2.

Maher, M. L., Simoff, S., Gu, N. and Lau, K. H. (2000) Designing virtual architecture, *Proceeding of CAADRIA2000*, pp. 481-490.

Reffat, R. and Gero, J. S. (2000a) Computational situated learning in design, *in* Gero, J. S. (ed.), *Artificial Intelligence in Design'00*, Kluwer, Dordrecht , pp. 589-610.

Reffat, R. and Gero, J. S. (2000b) Towards active support systems for architectural designing, *in* D. Donath (ed.), *eCAAD*e18, Bauhaus-Universitat, Weimar, pp. 143-147.

Saunders, R and Gero, JS (2001) A curious design agent, *in* JS Gero, S Chase and M Rosenman(eds), *CAADRIA'01*, Key Centre of Design Computing and Cognition, University of Sydney, pp. 345-350.

Suwa, M., Gero, J. S. and Purcell, T. (2000) Unexpected discoveries and s-inventions of design requirements: Important vehicles for a design process, Design Studies **21**(6): 539-567.

Tang, H and Gero, JS (2001) Roles of knowledge while designing and implications for CAAD systems, *in* JS Gero, S Chase and M Rosenman (eds), *CAADRIA'01*, Key Centre of Design Computing and Cognition, University of Sydney, 2001, pp. 81-89.