https://doi.org/10.7577/formakademisk.3077

Christos Chantzaras

Architecture as a system and innovation design discipline

A retrospective on architectural programming and its implications for the strategic extension of the discipline of architecture

Abstract

Talking about architecture means talking not only about buildings but also about processes or systems. In the latter context, architecture is a way of thinking and looking at people, spaces, interrelations and interactions. Proclaimed by IDEO's Tim Brown as one of the best system design forms of education available, architecture has potential in fields beyond the physical. In keeping with the views of renowned systems thinker Russell Ackoff, who graduated in architecture before focusing on operations research, the question arises whether the skills of architects can be applied more broadly in system and innovation design. This paper describes how architects deal with context and complexity from the perspective of the practice-oriented architectural programming method. From its early days in the 1960s, it offered architects a viable basis for an applied architectural design thinking method, but did not receive widespread attention from practitioners and academics. The method is critically assessed and compared to the known forms of design thinking from the viewpoint of industrial design. By describing a real-life project and students' work from a newly created seminar in a department of architecture, the paper investigates the current and future relevance of an advanced version of architectural programming for architectural practice and education. It stresses the desirability of reinforcing the core skills of architects by developing a design thinking method rooted in architecture, which needs to be taught, developed and disseminated. In the long term, it is argued, architecture should be considered and integrated as a 'systems and innovation design discipline' in the fields of systems thinking and innovation research.

***Keywords*:** innovation design, architectural programming, architectural design thinking

Working with context and complexity

Common conceptions of architects and architecture relate to the planning and design of buildings or physical structures. While industrial design disciplines have extended their field of influence, architecture has remained mostly in its traditional realm (Luebkeman 2015). Specific skills of architects, as well as architectural training, can be considered as viable bases for systems thinking and be transferred to systems analysis and design. Architects' understanding of systems is, in the words of Russell Ackoff, the best among all professions, but has been mostly pooled with other design disciplines and not explicitly described (Ackoff 1994; Hershberger 1999, pp. xvii-xviii; Burke & Tierney 2007; Lawson 2005; Dorst & Lawson 2009; Cross 2013; Fisher 2015b). The skills of architects are ideally inculcated in higher education and honed through project work in practice. Apart from varying quality and performance in reality, the skills are distinct from other design disciplines and need to be outlined separately. Architects abstract and reframe socio-technical complexities to a workable degree (Burke & Tierney 2007; Samuel 2018). They process obtained information on a problem visually and critically reflect in action on the state of their work (Schön 1983; Rowe 1987; Lawson 2005; Dorst & Lawson 2009; Cross 2013). Their "problem-solving mentality" looks "beyond the obvious for the profound – for understanding, not just description" (Hershberger 1999, pp. xvii-xviii). Architects deeply understand context, interactions and interrelations, and empathically

immerse themselves in a long-term perspective for their clients, while considering the changing usage of space over time (Nelson & Stolterman 2012; Pallasmaa 2016; Samuel 2018). They are constantly moving between problem and solution while analysing, synthesising and evaluating their approach, in order to create a state that "ought to be" (Lawson 2005, p. 49; Nelson & Stolterman 2012). They apply their non-linear thinking to handling wicked problems, using a broad range of methods and tools such as sketching, diagramming, modelling, prototyping and parametrising (Rowe 1987; Dorst & Lawson 2009; Gänshirt 2012; Cross 2013; Schumacher 2016). Furthermore, the architects' ability for synthesis relates to the real (world) and seeks ultimate application. This means that architects are used to constantly thinking at different scales from abstract diagrams, from drawings at 1:1000 to real solutions executed at 1:1. Prototyping in the case of architecture is the final result, and in most cases not a pre-state for scalable production. The models made prior to execution are representations of a final building or parts of it, but not the building itself. An architect has always to imagine, foresee and think of the implications and consequences when a design is going to be built. He has "no right to be wrong", as Rittel and Weber pointed out, and must retain a holistic perspective (Rittel & Weber 1984, p. 143; Dorst & Lawson 2009; Mäscher 2018). The action-oriented work of architects to create a new reality is combined with a systems-thinking approach for a larger context (Jones 2017; Keeley et al. 2013, p. 116). Finally, architecture is a social process, depending on interaction and collaboration with engineers, consultants and authorities. In this regard, architecture acts as a cross-discipline integrating different viewpoints into a new whole (Gharajedaghi 2011; Carraher et al. 2018).

In the starting phases of building projects especially, when vision, goals, requirements and needs are not clearly specified by clients, architects can play a strategic role and develop alternative directions that do not necessarily require a physical solution (Cherry 1999; Koolhaas 2004). With their dynamic, interactive and iterative design approach, architects can offer disciplines as management new approaches in the early phases for dealing with complexities, uncertainties and wickedness (Boland & Collopy 2004, p. 4). "By treating a wicked problem as a tame problem, energy and resources are misdirected, resulting in solutions that not only are ineffective, but also can create more difficulty, because the approach used is an intervention that is, by necessity, inappropriately conceptualized" (Nelson & Stolterman 2012, p. 17). But as the scopes of projects in the built environment and their contents are mostly developed before architects become involved, it may happen that the real challenge is not addressed (Figure 1) (Deamer & Bernstein 2010; Czaja 2017). Uncovered needs and processes, different user perspectives and changing demands over time from the markets and society may become apparent during the design process and architectural work. The lack of recognition of reality is caused to some extent by architects themselves, focusing on a minimal share of projects and special typologies such as museums, offices or single houses, while most new buildings are realised by construction, development and real estate companies. Projects thoughtfully designed by architects count only for 5–10% of new buildings in the United States and for approximately 2% globally (Deamer & Bernstein 2010, p. 9; Czaja 2017).

To enter the decision-making zone and deal with wicked problems of planning projects in a structured way, architects have developed different approaches, which are rarely made explicit or shared (Cross 2013). One of the few practice-oriented and published methods for an efficient and effective client interaction is called architectural programming (Faatz 2009; Bachman 2012). With this method, architects bridge the communication gap between clients, other stakeholders and consultants to reframe the complexities of building tasks, creating a common understanding on vision and goals and developing a new content prior to the design and planning (Cherry 1999; Henn 2004; Faatz 2009; Bachmann 2012). In contrast to a scientific approach to complexity or the consideration of messy complexity, architectural programming focuses on ordered complexity (table 1). It pursues all project-relevant information in order to uncover the essential problem to be solved by design in an institutional or real-life assignment with a client (Bachman 2012; Peña & Parshall 2012).

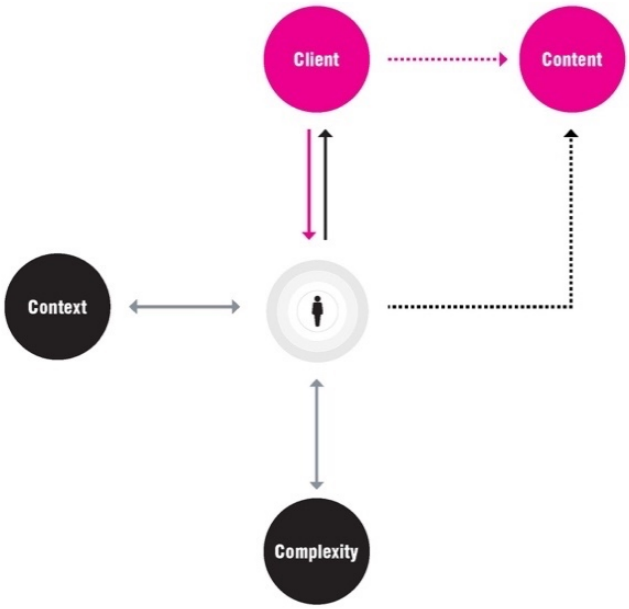

Figure 1. The architect communicates and engages with the client, while analysing the context and complexity of a project. The predetermination of the content by the client may lead to “bad” solutions, as the real challenge is not understood by design attitude (Boland & Collopy 2004, p. 4).

Table. 1. Complexity science and four modes of complexity as encountered by architecture (Bachman 2012, p. 74).

	Scientific	***Wicked***	***Messy***	***Ordered***	***Natural***
Proponents	Ackoff, Arnheim, Churchman	Simon, Rittel, Bell	Jacobs, Venturi, Alexander	Peña, Sanoff, Preiser	Geddes, Kepes, Olgyay
Postwar landmarks	1957	1957	1961	1965	1963
Realm	Cosmology	Society	Culture	Institution	Organisms
Objective	Adaptation	Intelligence	Spontaneity	Essence	Responsive
Organization	Dynamic	Cybernetic	Organic	Collaborative	Holistic
Order	Interactive	Managed	Authentic	Discovered	Emergent
Problem	Evolve	Bounded rationality	Identity	Reduce data	Flow
Agents	Systems	Decision makers	Citizens	Stakeholders	Systems
Application	Behavior	Organization	Urban	Definition	Design

Development of architectural programming

The term programming, i.e. setting up the project programme and brief, is familiar to architects to a greater or lesser extent. Architects do it unconsciously or consciously in the course of designing, in a phased way. As a structured and repeatable method applicable to a broad range of different tasks, architectural programming evolved in the late 1950s in the US, based on specific principles of action and understanding. Large, complex building projects required the integration of multiple stakeholders, as users, clients and clients’ consultants. When commissioned to design a high school facility, the architectural firm Caudill Rowlett Scott (CRS) developed a process model to structure the information, needs, requirements and goals

and simultaneously involve the stakeholders in a participatory way (Duerck 1993; Cherry 1999; Hodulak & Schramm 2011). The method helps raise the level of required information prior to the start of creating a building design by questioning the task, pursuing a shared vision for the project and creating a common understanding (Henn 2004; Faatz 2009). Regarded as a "research and decision-making process that defines the problem to be solved by design", architectural programming integrates elements of scientific research, project management and the architectural thinking process (Cherry 1999, p. 3; Duerck 1993). This process follows a strict separation of problem statement and solution, in order to avoid "trial-and-error design alternatives" and to prompt a rational architectural way of working: "Programming precedes design just as analysis precedes synthesis" (Peña & Parshall 2012, p. 10). Additionally, architects can apply methods and tools to analyse and design processes and systems beyond the physical. "The goal of architectural programming is to define problems to be solved by design. There should not be an underlying assumption that the solution must be the design of a building, and only a building" (Cherry 1999, pp. 229f).

In the decades since CRS (later CRSS) was established, the method was developed further in varying forms by HOK and other architectural practices. It became listed as an additional service by the American Institute for Architects (AIA), and was synonymously known as pre-design, brief design or facility programming (Duerck 1993; Kumlin 1995; Cherry 1999; Faatz 2009; Hodulak & Schramm 2011). In Germany it is mostly perceived as requirement planning and partially reflected in additional preliminary designs linked to honorary fee tables for architects and engineers (HOAI) or specific German standards and specifications (DIN). Few practices implement programming as a conceptual and integrative approach with architectural design (Henn 2004).

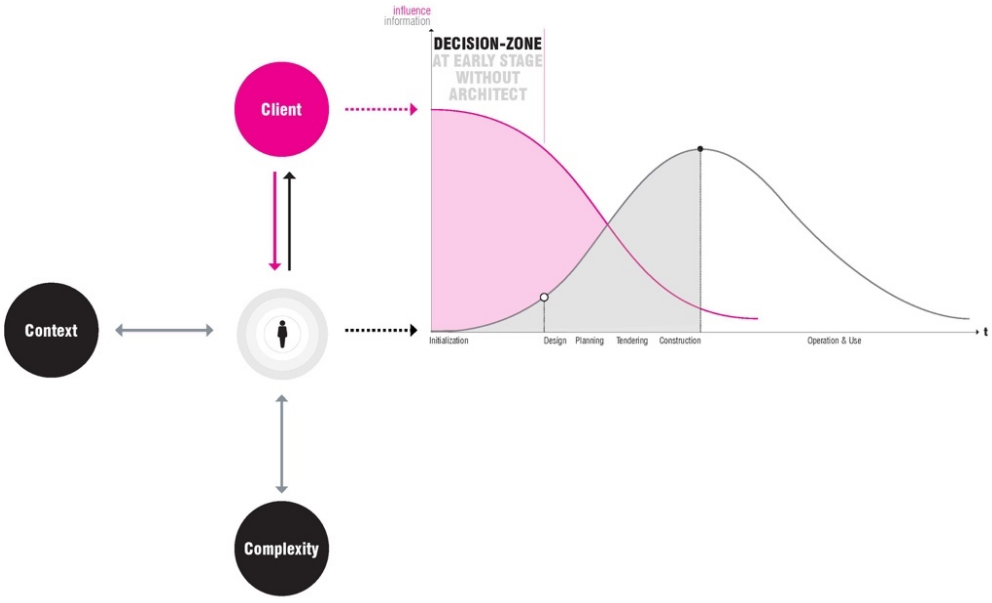

Figure 2. Normal distribution and progression of information and its influence on a building project. Architects approach the early-decision zone mostly within a smaller decision zone area (grey). (Own representation, drawn from Faatz (2009), p. 82; Henn (2004), p. 42.)

With architectural programming, architects act internally and externally in a co-creative and integrative way. Internally, architectural programming supports clarification and understanding for the design and planning team, which in many cases consists of further consultants, engineers and contractors or construction companies. Externally, in the interaction with clients and other stakeholders, the method allows architects to structure complex tasks, to order and separate information and to communicate in a comprehensible way with design- or architecture-distant

disciplines. By uncovering project-critical questions, architects can step into an early stage of decision-making and engage with clients and stakeholders, where the influence on the project's direction is high and the goals as well as the content of a project have not yet been identified (Figure 2 and 3).

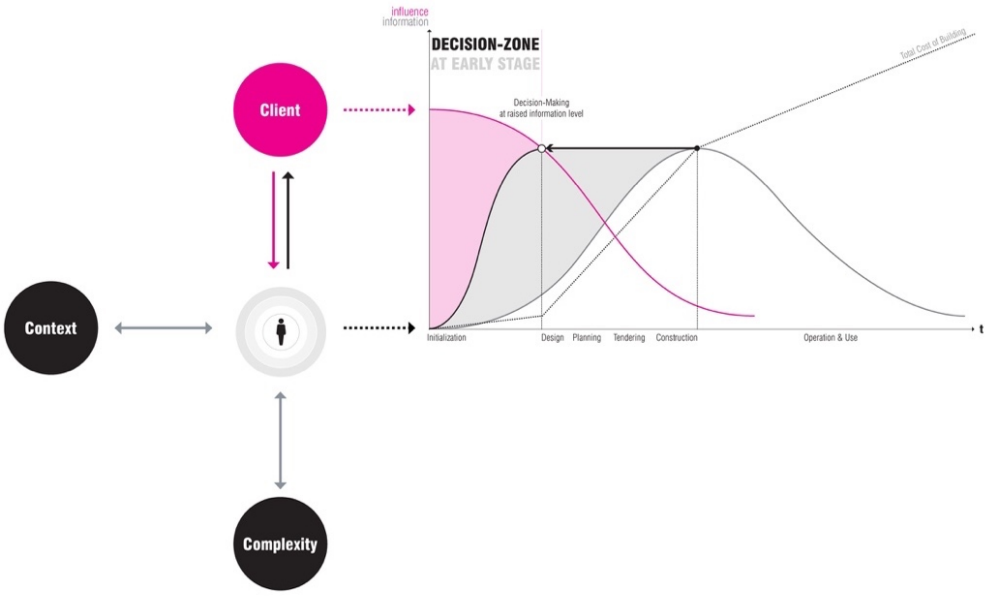

Figure 3. Applying architectural programming allows the information level to be raised before building design starts and offers architects greater access to the decision zone area (grey). (Own representation, drawn from Faatz (2009), p. 82; Henn (2004), p. 42.)

Problem seeking – basic process and principles

Fundamental to architectural programming is the process of staging in six to eight steps (Sanoff 1977). The starting phase entails clarifying the research to be undertaken, along with stakeholder co-ordination and project schedule set-up. It is followed by a preliminary workshop and an interview phase to clarify the vision and establish goals. An intense period of collecting and analysing facts and determining needs and requirements constitute, respectively, steps three and four (Peña & Parshall 2012; Cherry 1999; Hodulak & Schramm 2011). After having structured the information gained through research, interviews, workshops and submitted data, the concept phase begins. In this stage, graphic representations, diagrammatic sketches and models are created on a systemic level, and are compared with the initially developed vision, goals and needs. The final stage of the process is the statement of the problem or challenge that has to be solved by design. The steps or phases can be placed in a different order according to the needs of the project and on the basis of the high involvement and participation of the client, user and other stakeholders (Figure 4) (Peña & Parshall 2012; Cherry 1999).

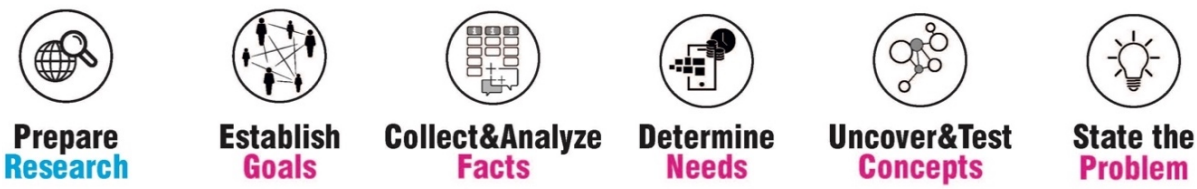

Figure 4. The basic process according to Peña & Parshall (2012, p. 2) and Cherry (1999). (Own representation.)

By describing this programming process as "problem seeking", the initial project statements by clients are questioned and challenged. The architects, called programmers in this context, are separating the analysis strictly from a synthesis through architectural design (Peña & Parshall 2012). The programmers follow a set of principles which are vital for successful understanding and reframing. "Programming is the definitional stage of design – the time to discover the nature of the design problem, rather than the nature of the design solution" (Hershberger 1999, p. 1). Further principles the programming architects consider are continuous client involvement, effective communication through visual and graphical depiction and a comprehensive structuring of data and information. The programmers apply an abstract, holistic form of systems thinking, conceptualise processes and the arrangement of functions and operate efficiently with iteration and loops to confirm the results and agreements of each step of the process. They focus on quantitative as well as qualitative information and precisely record, calculate and forecast demands in terms of square metres and areas, as well as collecting and listing the kind of technical equipment needed for the project and the mechanical, electrical and climate requirements (Peña & Parshall 2012; Cherry 1999).

Critical reflection in comparison with Design thinking

The rational approach of architectural programming helps to structure complex building projects, while retaining a holistic view on vision and context. The outlined ideal process of architectural programming requires, on the one hand, the client's commitment to proceeding through the steps of the project and, on the other hand, skilled, experienced and method-trained architects. Though methods and principles have been integrated and altered in practice, only a few architectural offices are explicitly offering programming as a service. One major criticism has been the reliance on organising, collecting and categorising the information and the strict separation between programming architect and design architect, which impedes a continuous overall design process and especially the iteration and learning loops needed for dealing with wicked tasks (Harrigan & Neel 1996; Noenning 2007; Peña & Parshall 2012). As the nature of the problem reveals itself over time and circumstances perhaps change, the division between architects as programmers handing over a project brief to architects as designers marks a break in team performance and a possible loss of design knowledge. The method remains a means to the end of planning functional buildings, without being developed further in its potential application to tasks beyond physical solutions. The rigid and formalised way of obtaining and visualising information does not evolve into a dynamic system design approach, although the graphical depiction of space, functions and stakeholder interaction provide a promising basis. Focused on frameworks and matrix analysis, architectural programming is seen more as a method of moderating and communicating with clients to clarify quantitative requirement planning, instead of an integrating concept for systems and design thinking. In the quantitative realm of facility management, it is framed in Germany in DIN 18205 "Brief for Building Design" and in ISO 9699 "Performance standards in building – Checklist for briefing – Contents of brief for building design" (Hodulak & Schramm 2011). The agile, systemic approach of architectural programming is not conveyed in this. Accordingly, only a small number of courses are available in higher education architectural faculties. What little consideration had been given to architectural programming in academic circles and in practice, which happened mostly in the US and the UK, petered out around the millennium. With few exceptions, architectural programming received minor attention thereafter among architects, students and researchers in Europe (Henn 2004; Faatz 2009). The principles and process steps of architectural programming are pursued more freely and in a constant transition between problem and design space (Henn 2004). With rising possibilities offered by digital tools and the increasing emphasis on other design thinking methods, architects have started to adapt their approaches for dealing with complex tasks. However, while design thinking as a method for creative innovation and collaboration has grown in importance and acceptance among diverse industries and management disciplines since the 1990s, awareness of architectural program-

ming and its experience in built projects, principles and process steps has dimmed (Buchanan 1992; Martin 2009). Surprisingly, architectural practices started to look at design thinking methods rooted in industrial design instead of leveraging the design thinking basis within their own discipline's body of knowledge. Comparing architectural programming with the design thinking method rooted in industrial design by its applied version for management and innovation processes reveals the existing knowledge (see Table 2).

Table 2. The architectural programming process, principles and skills, compared to industrial design thinking and extended by architectural system design. (Own representation after Cherry (1999), Lawson (2005), Lawson & Dorst (2009), Peña & Parshall (2012), Ambrose & Harris (2010) and Lewrick et al. (2017).)

		(Industrial) Design Thinking Method	**Architectural Programming Method**	**Programming Principles**	**Architect's Applied Skills**	**Level of Detail**
		research	prepare research	separation client involvement	questioning interviewing	**Abstract**
		understand	establish goals collect facts analyse facts	effective communication comprehensive structure abstract & system thinking	understanding formulating integrating interpreting	
		observe define	determine needs	efficient operation iteration & feedback	reframing sketching	
		ideate	uncover concepts test concepts	quantitative information qualitative information	layouting diagramming modelling parametrising moving forward evaluating	
			state the problem	problem resolution	synthesising	
	+		**Architectural System Design**			
		prototype	design system integrate requirements		designing layouting drawing planning parametrising	
			model build		modelling managing building	
		test	evaluate show		reflecting presenting	
		implement	design solution		applying	
	=		**Architectural Design Thinking Method**			**Concrete**

Architectural programming provides a viable basis for architects to interact at early stages – in Phase 0 – with clients and apply architectural skills for the understanding and design of strategic contents and systems. If a stronger integration with the architectural design process is to be fostered, an applied architectural design thinking method can be developed, which then can be independently practised for designing systems and innovations beyond a building context. The

structured process of industrial design thinking applied as fast-track innovation or a co-creation method in business environments is facing criticism on other grounds. By expelling intuition and aesthetics from its stages, it loses the innovative drive to contemplate the unexpected (Nelson & Stolterman 2012, p. 132; Verganti 2017). Yet, an architectural design thinking method can maintain the designer's freedom and agility to construct the way towards a solution (Buchanan 1992; Verganti 2017). As Lawson suggests, "good designers tend to be at ease with the lack of resolution of their ideas for most of the design process" (Lawson 2005, p. 154). This is part of the magic in the process, when the parts come finally together in the end and reveal the resilience architects may display when controverting or confronting the client's wishes, all in the cause of reaching a better state (Lawson 2005; Gänshirt 2012). The problem-seeking attitude that manifests itself in continuous questioning of the existing could be beneficial to innovation design (Figure5). The thinking of architects in a larger context and over longer periods of time is leading beyond obvious or short-term solutions. A quote dating back to English architect Bryan Lasdun explains what the job of an architect is: "Our job is to give the client [...] not what he wants, but what he never dreamed he wanted; and when he gets it, he recognizes it as something he wanted all the time" (Cross 2007, p. 52).

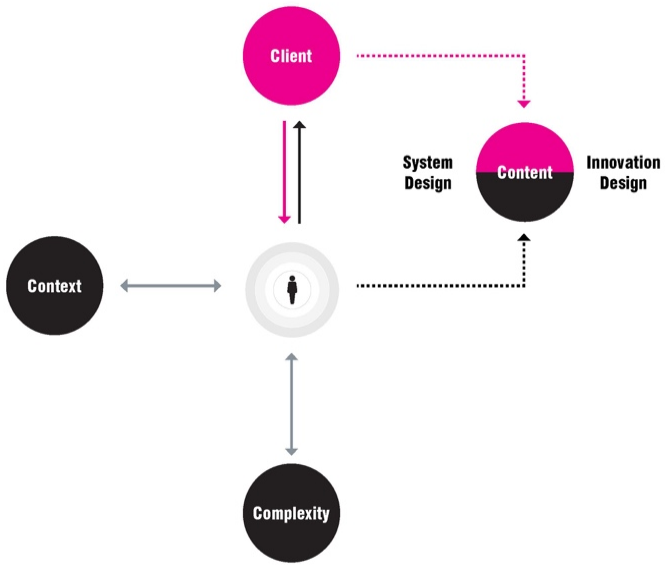

Figure 5. Content creation in integrative and co-creative work between client and architect, consisting of system design (ideation and invention) and innovation design (implementation).

Architectural programming in practice

An architectural office is still commonly conceived of as characterised by informal, less transparent or less structured ways of working. Few offices integrate the method and communicate it explicitly to clients or third parties. Their scope of work ranges from strategic to building design, as can be seen in the services of the global architectural firms of HOK or Gensler. In 2015, the author conducted a study on one example from practice, the architectural programming project for a research & development headquarter in nanotechnology. The case study description and analysis are based on the author's notes and reports, presentations and models handed over by the company (attocube & Henn 2015).

The nanotechnology start-up, founded in 2001, was in a transition phase from a grown structure of 90 people to a company of more than 175 employees. The main questions in 2015 were: how to grow as a company while maintaining the spirit of a community and start-up; how to integrate management departments with innovation, research and highest-precision production; and how to communicate a mindset and values and remain attractive to researchers

and high-potential employees. The research followed the programming process through several steps: (1) goal definition through workshops and interviews, (2) collection of facts and requirements, (3) the determination of needs, (4) conceptualisation of new work processes, and (5) the modelling of a system for the new building as a statement to be transformed into a building design (6).

Through architectural programming, the company reflected their vision, values and processes. In graphical depicted and recorded interviews, the core of the company's activities was made explicit: first, to serve human beings with products from investigating space to analyse matter beyond human recognition; and second, to focus on the employee 1:1 as the source of creativity and innovation beyond hierarchical settings. The visualisation created after several iterations became elemental for the final system design. It integrated the scales in which the products of the company are operating, with the cycle of life connecting theses scales and the purpose of the company to actually produce what has been theoretically thought (Figure 6). Making the invisible visible resulted in a statement of the vision, spirit and innovation of the company.

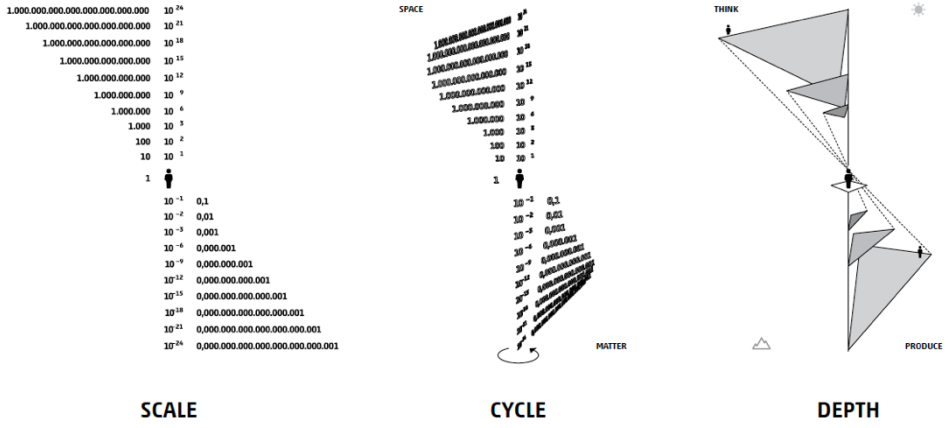

Figure 6. Scale, cycle and depth describe the contexts in which the company is located. Central to all endeavours, inventions and developments is the human being, as user and employee (Source: attocube systems & Henn (2015) and Henn (2016)).

The available reports and documents on the company were processed visually by sketches, drawings, parametric simulations and physical models and enriched by the information gained throughout the workshops and interviews with the executive board, researchers, developers and administrative staff. The process of business and production was mapped in plan views and sections and brought together in a three-dimensional diagram of the organisation. Employees, distance and proximity, as well as process allocation and spatial functions, were considered in the model. The digitally rendered visualisation was then viewed from different angles and revised in order to show the flows of interaction (Figure 7).

By cutting through the model, a cross-section of the entire company structure was created, which additionally displayed the values and mindset of the company. The diagrammatic section served as a structure by which to translate between narrative and organisation. In extruding the section in different directions, the innovation process space evolved, balancing out awareness, functions and project flows of the company (Figure 8).

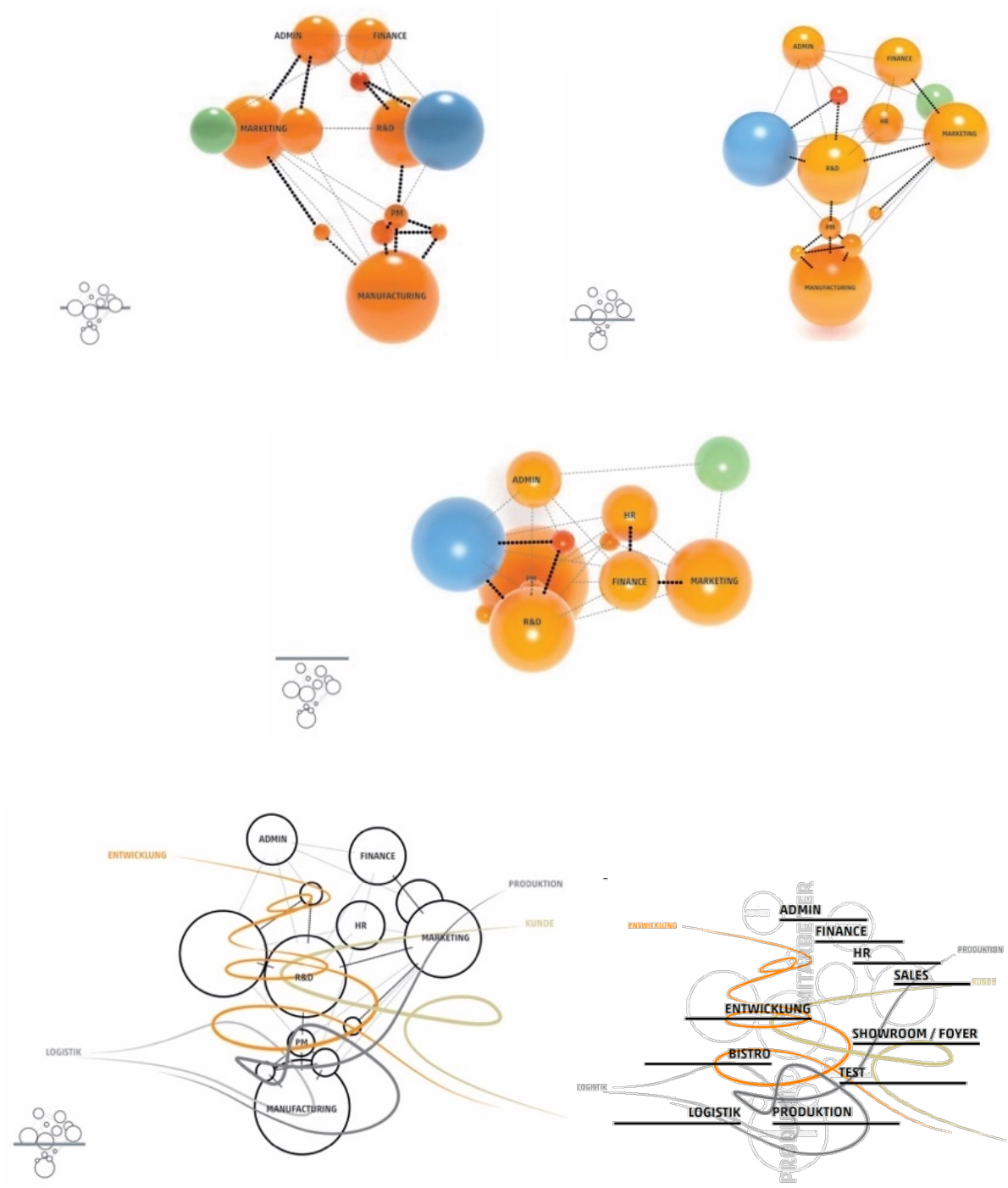

Figure 7. Three-dimensional model of employees, relationships and spatial organisation (Source: attocube systems & Henn (2015) and Henn (2016)).

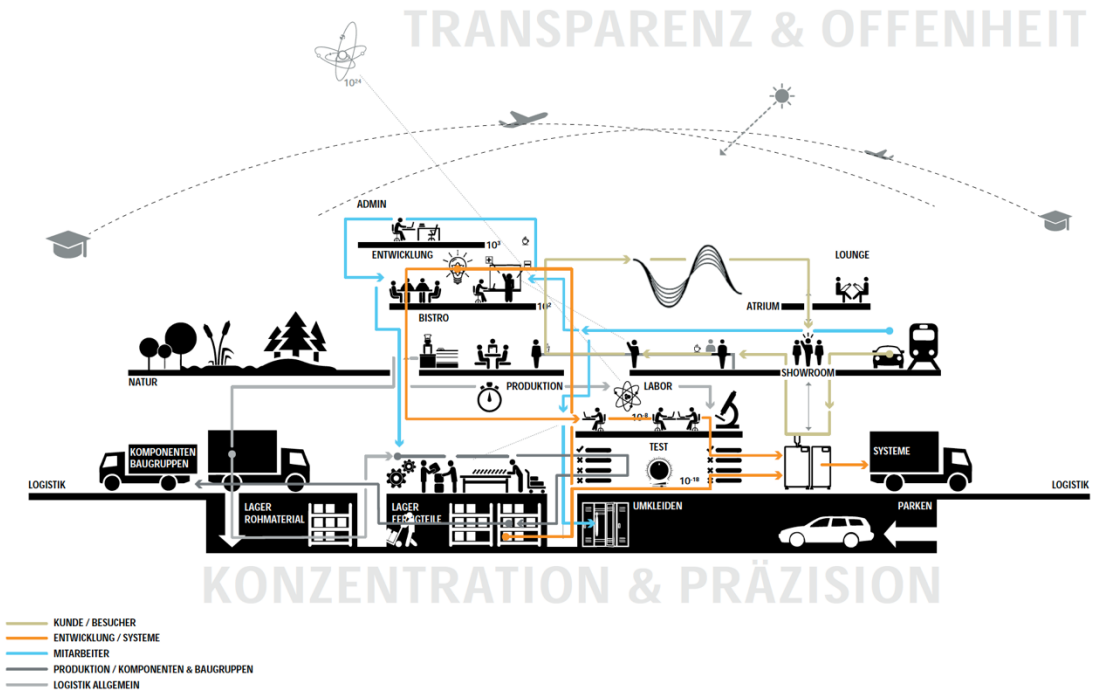

Figure 8. Concept diagram section of values, employees, relationships and spatial organisation (Source: attocube systems & Henn (2015) and Henn (2016)).

Architectural programming and Design thinking in higher education

Despite its potential for complex problem-solving, architectural programming is rarely taught in architecture faculties in European institutions. In the Winter semester 2016/17, the Department of Architecture of Technical Univeristy of Munich started a course to foster awareness of the relevance of new approaches at the intersection of client, architect and society. The main goal is to leverage the potential of architectural thinking and its tools to fields beyond the physical and empower students to communicate, apply and integrate their expertise and creativity in new areas and widen the perspective of architectural education. The course took architectural programming as the basic method, but continued as architectural design thinking in Phase 0, to overcome the building focus and educate architects on their future roles in facing the challenges of the built environment, such as mobility, work, innovation, construction, resources and consumption. It outlines the opportunity for architecture and architectural education to engage with questions in management, economics, sociology, information technology and data sciences and develop itself as a system and innovation design discipline (Boland & Collopy 2004; Shamiyeh 2007; Burke & Tierney 2007; Hyde 2012; Fisher 2015b; Luebkeman 2015). Thomas Fisher concluded that, “by 2050, leadership could become one of the most recognized and well-rewarded skills that architects have to offer” (Fisher 2015a, p. 45).

The students of the course define their individual task in four iterative steps: understanding context and complexity, reframing the problem, developing a concept and designing a system. The following projects emerged in the next semesters, showing how architectural students have applied their talents and skills to new fields of interest. For the students, the absence of a stated problem in the beginning was unusual and difficult to cope with. In contrast to the traditional architectural programming courses, which were teaching the method in relation to a real or fictitious building project, the students in the course in question were asked to discover a problem and to solve it by themselves.

Mobility Guide – Interactive system for personalised mobility / Andreas Beigel

This student's project (Figure 9) started by analysing modes of transport and the future of mobility. It soon led to the question, why the modes of transport are not individually chosen according to an individual set of criteria and preferences. The existing offers seemed to be mostly supply-driven and not demand-driven, without a reliable comparison of sustainability issues or personal trade-off options. This led to a second consideration on a systems level, where an individual's decision could influence the decisions of others. In the future, it is hypothesised that transport will not be a single transport vehicle but a user-centred and individually generated transport system. The system concept for a mobile application to serve as a mobility guide was researched further in the student's master thesis.

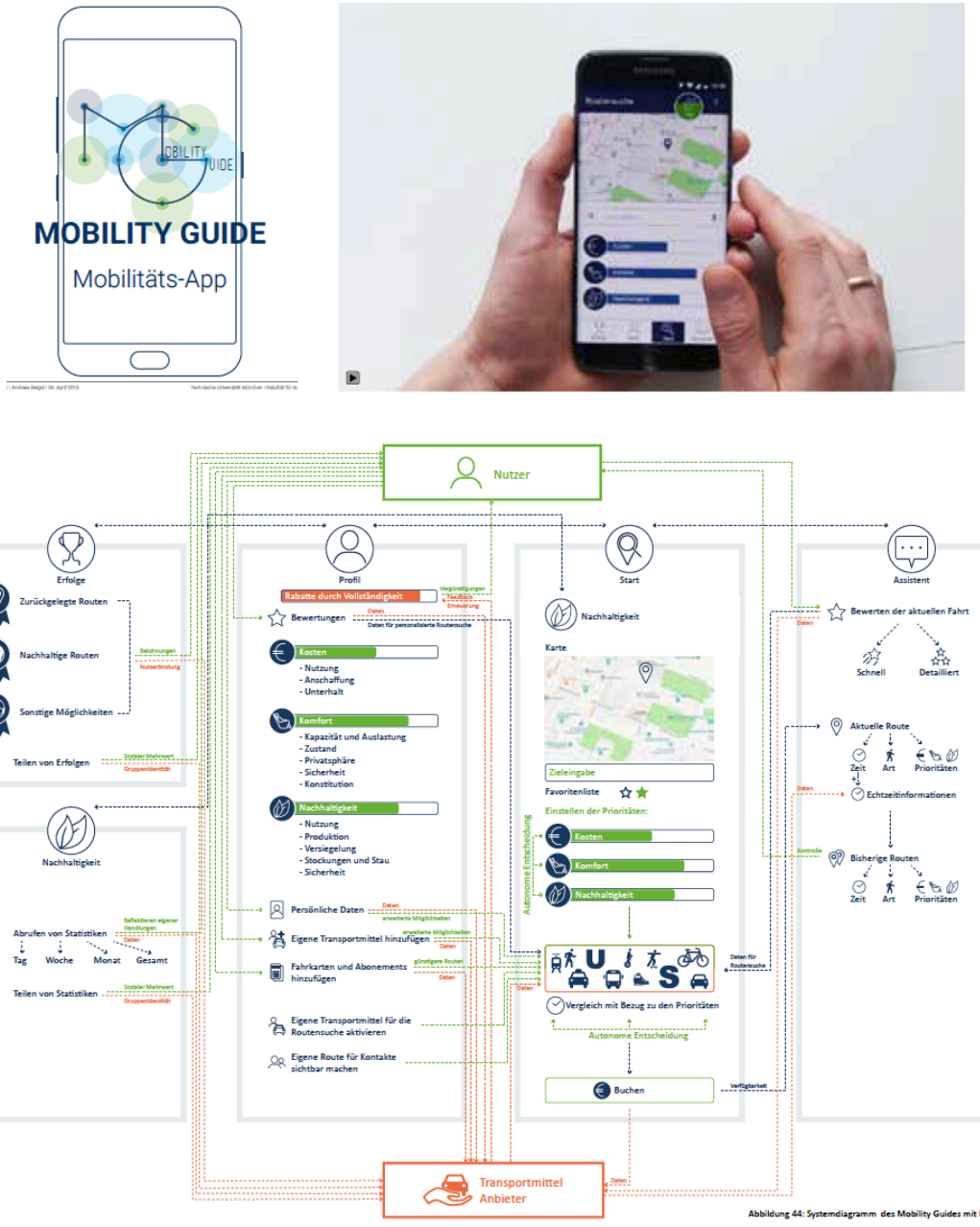

Figure 9. Application architecture for an individual mobility guide with a set of criteria for the selection of the individual transport mode. (Visualisation by Andreas Beigel (2017), p. 89 and acc. presentation.)

Bringing Design thinking to school / Sara Hozzánková

Questioning the sources of misconceptions and miscommunication between architects and laypeople, one student discovered a crucial gap in early design education. Although children are taught mathematics, languages, geography and later on physics, chemistry and other subjects, design is neglected in school. The student's work visualises, in a playful way, what keeps neighbourhoods of thoughts and education apart, and shows a way to achieve better understanding (Figure 10). By nurturing design thinking in school, especially for use in the field of architecture, the children would better understand this way of thinking and collaborate as grown-ups in the development of a liveable built environment.

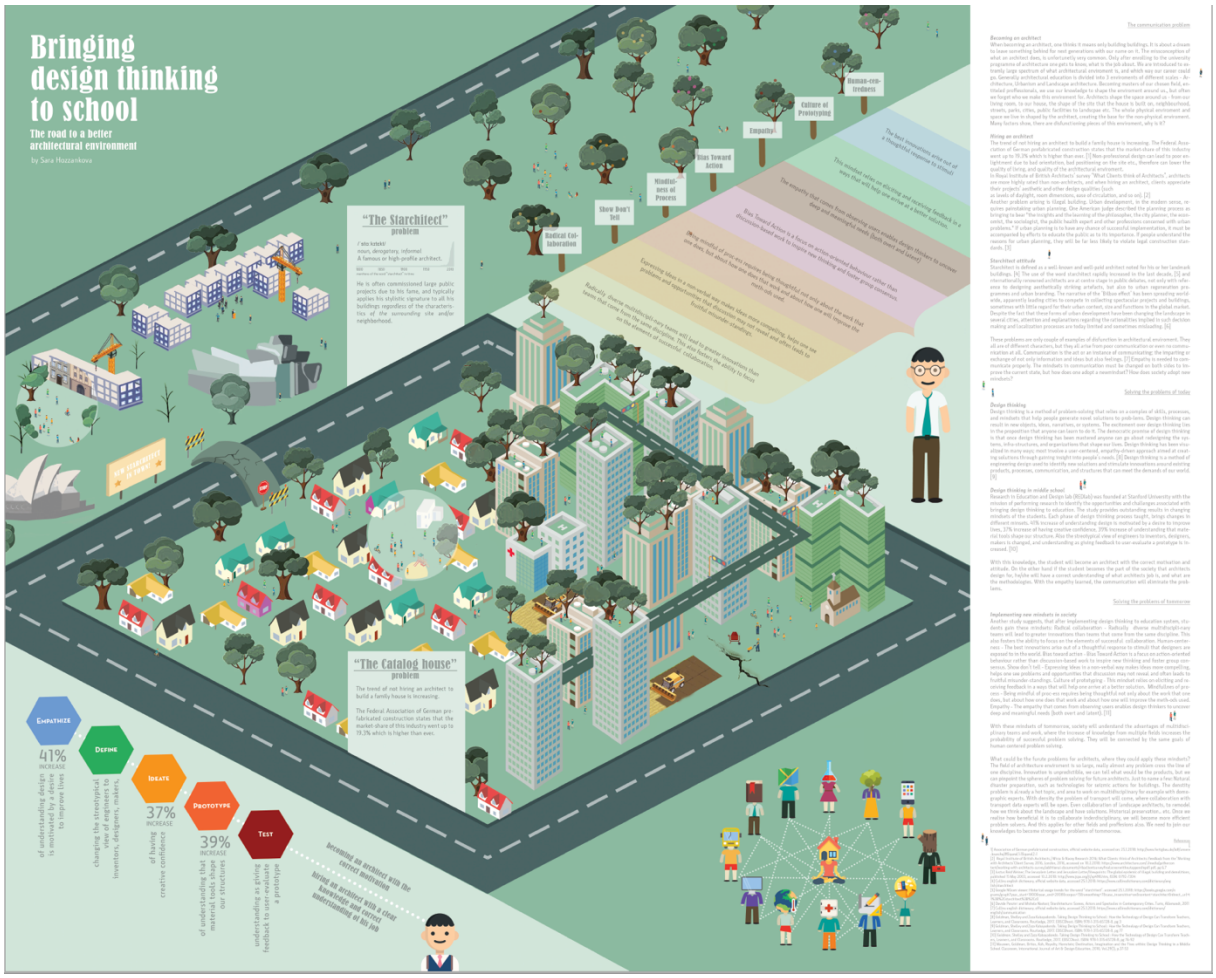

Figure 10. Bringing design thinking to school – map explaining misunderstandings and misconceptions on design and architecture and offering playful roads to a better architectural understanding. Visualisation by Sara Hozzánková.

Liveable Smartness - A strategic Proposal for Architects to improve. Liveability through Smart City Implementations / Yonne-Luca Hack

After laying the foundations during the course for synthesizing liveability and smart city concepts, the student analysed in his master thesis about 200 scientific papers, screened the most successful tech companies, evaluated existing city concepts and extracted contributions from architectural magazines. In his results he revealed a strong focus on technology and organization, and a non-focus on space. He subsequently conceptualized an integrative approach of categories considering equally the social, the organizational and the spatial realm. He concluded with a strategic proposal for architects to improve liveability through smart city

implementations, and re-integrate themselves as actors in Phase 0 for the future planning of urban settings (Figure 11).

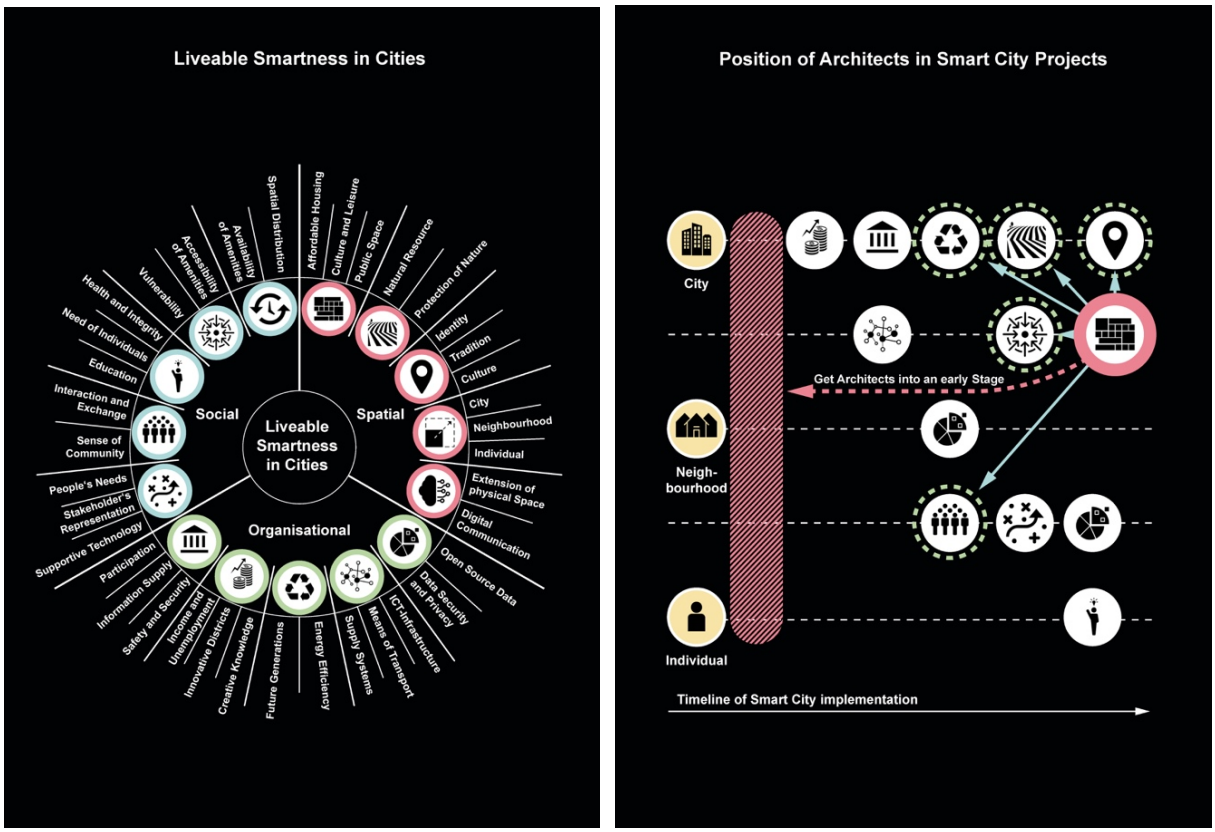

Figure 11. The Liveable Smartness model as a three-layered classification tool by balancing social, organizational and spatial dimensions. In a potential smart city implementation process the late integration of architects could be transformed by shifting towards a mind-set of system thinking in digital space. (Visualisation and description by Yonne-Luca Hack (2019), p. 56-71.)

Outlook and further steps

Architects have their specific way of design thinking and are capable of extending it into systems thinking and system and innovation design. The links to real life, the laying of foundations at a human scale and a tangible understanding of human interactions, paired with creative, abstract and diagrammatic thinking, are viable means of entering new fields. But for architects to be aware of these skills and ways to act on them they need training, both in academia and in practice. The present paper is a first approach to raising awareness of existing methods such as architectural programming and its further integration into higher education. The deficiencies of the ideal programming process explained here need to be addressed by comparing it with other successfully applied design thinking methods (e.g. design sprints, scrums or hackathons) and considering new ways and tools for collaboration and co-creation. To elaborate a concept for architectural design thinking as a method, research and exchange with other design disciplines will be necessary. The present evaluation of the courses observed yields important insights into the opportunities for adjustments and additions. The overall rate of satisfaction by the participants in the courses was very high, based on the anonymous evaluation conducted by the Department of Architecture. It was extremely valuable for the students to have the opportunity to develop questions and a topic independent of building constraints and in a structured way. Learning to communicate complex problems as well as relating them to a larger context was important for the students, and encouraged them to think

of future roles for architects in the fields of mobility, resources, digitisation, smart cities, construction and work. This is of major importance, as the primary focus of architectural education and practice is still building design, which represents only a small share of new constructed projects around the world. By leaving aside the building plan, the potential of architectural education and practice could be made accessible to other fields. This direction requires an understanding and training in the principles and process steps in order to engage with other disciplines and develop common projects or research questions. In the long term, architecture could become not merely building design but also a system and innovation design discipline.

Christos Chantzaras
Research Associate
Department of Architecture, Chair of Architectural Informatics - Prof. Dr. Frank Petzold
Technical University of Munich
christos.chantzaras@tum.de

References

Ackoff, R. (1994). If Russ Ackoff had given a TED Talk... Presentation from a 1994 event hosted by Clare Crawford-Mason and Lloyd Dobyns to capture the Learning and Legacy of Dr. W. Edwards Deming. Retrieved March 2, 2019, from https://www.youtube.com/watch?v=OqEeIG8aPPk

Ambrose, G., Harris, P. (2015). *Design thinking for visual communication* (2nd ed.). London, New York: Fairchild Books, an imprint of Bloomsbury Publishing (Basics design).

Attocube systems, & Henn (2015): Programming for Attocube systems AG. Internal Report. Unpublished Excerpts released for publication.

Bachman, L. R. (2012). *Two spheres. Physical and strategic design in architecture*. London: Routledge.

Beigel, A. (2018). *MOBILITY GUIDE - Interaktives System für die personalisierte Mobilität. (*Master thesis) Department of Architecture, Technical University of Munich. Unpublished.

Buchanan, R. (1992). Wicked problems in design thinking. *Design Issues*, *8*(2), 5-21.

Burke, A., Tierney, T. (2007). *Network practices – New strategies in architecture and design*. New York: Princeton Architectural Press.

Boland, R. J., Collopy, F. L. (2004). *Managing as designing*. Stanford, CT: Stanford University Press.

Camillus, J. C. (2008). Strategy as a wicked problem. *Harvard Business Review* (Strategic Planning, May 2008). Available online at https://hbr.org/2008/05/strategy-as-a-wicked-problem, updated on 6/5/2018, checked on 6/5/2018.

Carraher, E., Smith, R. E. & DeLisle, P. (2017). *Leading collaborative architectural practice*. Hoboken, NJ: John Wiley & Sons.

Cherry, E. (1999). *Programming for design*. New York: Wiley.

Cross, N. (2007). *Designerly ways of knowing*. Basel: Birkhäuser.

Cross, N. (2013). *Design thinking. Understanding how designers think and work* (Reprinted ed.). London: Bloomsbury Academic.

Czaja, F. (2017). Interview. Architekt Carlo Ratti: "Die Spezies Architekt wird aussterben." Retrieved November 3, 2017, from *Der Standard* http://derstandard.at/2000058656475/Architekt-Carlo-Ratti-Die-Spezies-Architekt-wird-aussterben?ref=article

Dubberly, H. & Rith, C. (2006). Why Horst W. J. Rittel matters. *Design Issues, 22*(4).

Duerk, D. P. (1993). *Architectural programming – Information management for design*. New York: John Wiley & Sons.

Deamer, P., Bernstein, P. G. (2010). *Building (in) the future: recasting labor in architecture*. New York: Princeton Architectural Press.

Faatz, S. (2009). Architectural programming: providing essential knowledge of project participants needs in the pre-design phase. *Organization, Technology and Management in Construction – an international journal, 1*(2).

Fisher, T. (2015a). Welcome to the Third Industrial Revolution. *AD Architectural Design,* (236, July-August), 40-45.

Fisher, T. (2015b). Labor and talent in architecture. In P. Deamer (Ed.), *The architect as worker. Immaterial labor, the creative class, and the politics of design*. London: Bloomsbury Academic.

Gänshirt, C. (2012). *Tools for ideas. Introduction to architectural design*. Basel: de Gruyter.

Gharajedaghi, J. (2011). *Systems thinking. Managing chaos and complexity: A platform for designing business architecture* (3rd ed.). Burlington: Elsevier Professional.

Harrigan, J. E., Neel, P. R. (1996). *The executive architect. Transforming designers into leaders*. New York: Wiley.

Hack, Y.-L. (2019). *Liveable Smartness - A strategic Proposal for Architects to improve Liveability through Smart City Implementations.* (Master thesis) Department of Architecture, Technical University of Munich. Unpublished.

Henn, G. (2004). Programming – Projekte effizient und effektiv entwickeln. In O. Schürer, G. Brandner, *Architetur: consulting* (pp. 42-49). Basel: Birkhäuser.

Henn (2016). Attocube systems Unternehmensitz, Haar/München, In Henn Yearbook 2016 (pp. 106-109). Munich: Henn.

Hershberger, R. G. (1999). *Architectural programming and predesign manager*. New York: McGraw-Hill.

Hodulak, M., Schramm, U. (2011). *Nutzerorientierte Bedarfsplanung Prozessqualität für nachhaltige Gebäude*. Heidelberg: Springer.

Jones, P. (2017). The systemic turn. Leverage for world changing. *She Ji. The Journal of Design, Economics, and Innovation 3*(3), 157–163. https://doi.org/10.1016/j.sheji.2017.11.001

Jones, P. H. (2014). Systemic design principles for complex social systems. In G. Metcalf (Ed.), *Social systems and design*. Translational Systems Sciences, Vol. 1. Tokyo: Springer: Tokyo.

Keeley, L., Pikkel, R., Quinn, B., Walters, H. (2013). *Ten types of innovation. The discipline of building breakthroughs*. Hoboken, NJ: Wiley.

Koolhaas, R. (2004). *Content*. Cologne: Taschen Verlag.

Lawson, B. (2005). *How designers think: The design process demystified* (3rd ed.) Oxford: Architectural Press.

Lawson, B., Dorst, K. (2009). *Design expertise*. Oxford: Elsevier Architectural Press.

Lewrick, M., Link, P., Leifer, L., Langensand, N. (2017). *Das Design Thinking Playbook. Mit traditionellen, aktuellen und zukünftigen Erfolgsfaktoren.* Zurich: Versus.

Luebkeman, C. (2015). Design is our answer – An interview with leading design thinker Tim Brown. *AD Architectural Design* (236, July-August), 34-39.

Mäscher, T. (2018). How architectural thinking and research collaboration brings value to creative industries. *Archipreneur Magazine* (1), 96–104.

Martin, R. L. (2009). *Design of Business. Why Design Thinking is the Next Competitive Advantage*. Boston: Harvard Business Review Press.

Peña, W. M., Parshall, S. A. (2012). *Problem seeking: An architectural programming primer* (5th ed.). Hoboken, NJ: Wiley.

Kumlin, R. M. (1995). *Architectural programming – Creative techniques for design professionals*. New York: McGraw-Hill.

Nelson, H. G. & Stolterman, E. (2012). *The design way – Intentional change in an unpredictable world* (2nd ed.). Cambridge, MA: MIT Press.

Noenning, J. (2006). *Architektur Sprache Komplexität. Acht Essays zur Architekturepistemologie*. (Doctor dissertation), Bauhaus-Universität, Weimar, Germany.

Pallasmaa, J. (2016). Spatial choreography and geometry of movement as the genesis of form. In M. Kanaani, D. A. Kopec (Eds.), *The Routledge companion for architecture design and practice* (pp. 35–44). London: Routledge.

Rittel, H. W. J., Webber, M. M. (1984). Planning problems are wicked problems (1973). In N. Cross (Ed.), *Developments in design methodology*. Chichester: Wiley.

Rowe, P. (1987). *Design thinking*. Cambridge, MA: MIT Press.

Samuel, F. (2018). *Why architects matter. Evidencing and communicating the value of architects*. Oxfordshire: Routledge.

Sanoff, H. (1977). *Methods of architectural programming*. Stroudsburg, PA: Dowden, Hutchinson & Ross.

Schön, D. A. (1983). *The reflective practitioner. How professionals think in action*. New York: Basic Books.

Schürer, O. & Brandner, G. (2004). *Architektur: consultin*g. Basel: Birkhäuser.

Schumacher, P. (2016). Parametricism 2.0. Rethinking architecture's agenda for the 21st century. *AD Architectural Design* (240, March-April).

Shamiyeh, M. (2007). *Organizing for change profession – Integrating architectural thinking in other fields*. Basel: Birkhäuser.

Verganti, R. (2017). Design thinkers think like managers. Two strategies linked to uncertainty resolution. *She Ji: The Journal of Design, Economics, and Innovation 3*(2), 100–102. https://doi.org/10.1016/j.sheji.2017.10.006