Conceptual Evolution of Architectural Program Through the Second Half of the 20th Century

Ülkü Özten
Eskişehir Osmangazi Üniversitesi
Mimarlık Bölümü

Öz

Makale mimari programın yirminci yüzyılın ikinci yarısı boyunca kavramsal evrimini izler. Mimari program nosyonunun mimarlık söylemindeki evrimini ve mimarlıktaki konumunu çeşitli kuramcıların farklı görüşler sunan çalışmalarında gözden geçirir. Analiz-sentez ve varsayım-yanlışlama olarak ifade edilen farklı iki ana paradigmatik epistemolojik pozisyon altında kategorize edilebilecek söz konusu yaklaşımları ilgili pozisyonlara referansla gözden geçirmek suretiyle tartışır.

Makale dört ana bölüm altında yapısallaştırılır. Durumu ve problemi özetleyen giriş bölümünü takiben ikinci bölümde ilk paradigmatik pozisyon olan analiz-sentezin epistemolojik kökenleri belirtilerek dönemin program anlayışının söz konusu epistemolojik kökene referansla nasıl kavramsallaştığının yanı sıra program-geleneksel tasarım, program-programlama-hesaplama gibi temel tartışma alanları, problemi ele alan farklı kuramcıların çalışmalarına referansla tartışılır. Üçüncü bölümde, bir karşı paradigma olarak gündeme gelen varsayım-yanlışlama modeli epistemolojik kökenleri üzerinden incelenir ve farklı kuramcıların çalışmalarında mimari programın nasıl değişip dönüştüğü; analiz-sentez modelinin eleştirisi üzerine kurulan varsayım-yanlışlama modeli altında mimari program anlayışının nasıl evrimleştiği araştırılarak tartışılır. Sonuç bölümünde ise, makale, mimari programın yeniden kavramsallaştırılmasına yönelik mimarlık gündemindeki güncel "durumu" ortaya koyar.

Anahtar Kelimeler
Mimari program, Analiz-Sentez, Varsayım-Yanlışlama, Mimari Epistemoloji

Summary

This paper traces the conceptual evolution of architectural program through the second half of the 20th century. It discusses the evolution of the notion and its place in the architectural discourse in the works of various scholars, while giving and reviewing essential views, with a particular focus on two paradigmatic epistemological positions under which these views could be categorized: analysis-synthesis and conjecture-refutations. Aiming towards a reconsideration of the notion, it concludes with presenting our present understanding and conception of the notion, namely the present "state" of architectural program in the architectural agenda.

Keywords:
Architectural program, Analysis-Synthesis, Conjecture-Refutation, Architectural epistemology

Acknowledgement: This article is based on the author's doctoral dissertation completed at the Middle East Technical University – Ankara, under the direction of Mine Özkar.

First, the taking in of scattered particulars under one Idea, so that everyone understands what is being talked about…Second, the separation of the Idea into parts, by dividing it at the joints, as the nature directs, not breaking any limb in half as a bad carver might.

—Plato, 265D

It is not hard to see why the analysis-synthesis, or inductive, notion of design was popular with theorisers and even with designers as a rationalisation of their own activities. The architectural version of the liberal-rational tradition was that designs should be derived from an analysis of the requirements of the users, rather than from the designer's preconceptions. It is directly analogous to the popularity of induction with scientists who were anxious to distinguish their theories as being derived facts in the real world.

—Hillier, 1972

1. Introduction

The discussion of *program* in the field of history and theory of the *Modern Movement* in architecture enters a new period in the 1960s and 70s under the influence of the strong embodiment of science and technology in the field of *design research.* Beginning with this period, the idea of program arises both in a series of generations parallel to a specialized, sophisticated understanding of design and as an autonomous professional area of study. While the concept was being largely developed under the influence of a scientistic, positivist epistemological framework called *analysis-synthesis*, since the mid-1960s, it had also become the focus of a parallel research campaign for a Popperian counter-framework: *conjecture-refutation*. As an alternative to analysis-synthesis, the conjecture-refutation framework addresses a series of issues and yields to new principles results in a reconceptualization of program.

2. Architectural Program from the Perspective of Analysis-Synthesis

In the Merriam Webster's dictionary *(1993, p.1812)*, the word "program" is first defined as "public notice" in relation to social performances and entertainment. Yet in a further definition, program is explained

as "a plan of procedure: a schedule or system under which action may be taken toward a desired goal". Then, in reference to Sullivan, a further definition was given as "a statement of an architectural problem and of the requirements to be met in offering a solution". It is in this sense that the word program penetrated vocabulary of the modern architecture in the early twentieth century.

On the other hand, starting from the early 1970s, architectural program has changed over from the traditional form-function discourses to a programming-oriented rhetoric. Basic references of today's architectural programming discourse were established along a line of thought initiated by a group of American researchers including Pena & Parshall *(1969)*, Sanoff *(1970)*, Duerk *(1993)*, Kumlin *(1995)*, Cherry *(1999)* and Hershberger *(1999)*.

The discourse is grounding on the argument that some of the worst examples of architecture and planning had happened in the period between 1945-1975 *(Kumlin, 1995, 1)*. In the broadest sense, the issue has evolved under the term "facility programming" on a methodological critique of the existing situation that there is a strong need to be systematic to overcome the faults of the earlier design. With its emphasis on the idea of economy as well as simplification of design, transparent and causal description of design process, programming was highly preferred by commercial enterprises. As such, it has emerged as an area of expertise in the service of building industry. Then it has become a professional research area operating a set of analytical studies on "establishing goals, collecting and analyzing facts, uncovering and testing concepts, determining needs and finally stating the problem" *(Pena & Parshall, 2001, 24)*.

Programming studies have emerged as an embodiment of analysis-synthesis. Following the footsteps of the *scientific method*, it leads researchers to focus on the "facts", or the details of the user needs by decomposing design problem into its elements. From this perspective, design is assumed as a problem operated in a two-fold process: problem seeking and problem solving. Pena and Parshall describe how the first one, programming, enables the second:

> *These are two distinct processes, requiring different attitudes, even different capabilities. Problem solving is a valid approach to design when, indeed, the design solution responds to the client's design problem. Only after a thorough search for pertinent information can the client's design problem be started. "Seek and you shall define!" (Pena & Parshall, 2001, 15).*

The functional division of programming from designing entails a clearly separated pre-design process where the aim is "searching for sufficient information to clarity, to understand and to state the problem" *(Pena & Parshall, 2001, 15)*.

2.1 Epistemological Background of Analysis-Synthesis

The ideas central to the conception of the growth of knowledge in the scientific method *(or induction- a method of obtaining knowledge through verification)* are mainly developed by Francis Bacon and Rene Descartes in the 17th century. Assuming the "highest" and "absolute" certainty *(Descartes, 1899)*, Descartes declares that "all the laws of nature follow with necessity from the one analytic principle" *(Popper, 2005, 451)*, and Bacon puts forward experimental and rational examinations of things as opposed to the prejudice-based "dogmatical" method *(Bacon, 1905, 76-77)*. Analysis-synthesis as the method of inductivist thought *(which is "the only correct mode of elaborating facts" (Windelband, 1901, p. 384)*) lays out "a theory of science which holds that scientific knowledge consists of the laws or principles derived by inductive canons from accumulated facts" *(Bunnin & Yu, 2004, p. 343)*. By rejecting the authority of custom as the source of knowledge, such position argues that intellectual capacity of the individuals has privileges over the scholasticism. The position has three main points. First, it argues that scientific enquiry begins with observation, or facts; second, it pursues the idea of a method of

discovery which conducts us in an orderly fashion, and third, it assumes that scientific theories are in nature intolerant to possible errors inherent into the process in general. Analysis-synthesis represents scientific discovery as an inductive rational process "where objective facts enter a passive mind and leave their traces there" *(Feyerabend, 1993, p. 152)*. Analysis-synthesis method or *naive inductivism* as called by Bamford is the common traditional view of scientific method. Starting from the former to the latter it draws a straight line between verification and theory *(Bamford, 2002, 246)*.

Program-biased perspectives in architecture, especially the ones that comes after the *Design Methods* movement[1], are usually put forth as vanguards of a scientific, progressive design research ideal and of objectivity against tradition of design history and theory. The reason for such radical positioning is based on the assumption that design must be systemmatic, rational and factual. The epistemological model in which the idea of program-biased architecture appears is called analysis-synthesis. The model mainly represents stages of Baconian analysis-synthesis as transformed to the design methodology *(Jones, 1963) (Archer, 1965) (Broadbent, 1969) (Bamford, 2002)*. Coyne and Snodgrass explain the model as a flow diagram where analysis -defining the problem- leads to a synthesis -seeking a solution- and the evaluation of the results feeds back in a loop *(Coyne & Snodgrass, 1995)*.

The following section will focus on the key figures and key discursive components of the lineage of inductive programming *(analysis-synthesis)* in architecture as started from the 1970s.

2.2 Programming as an Alternative to the Traditional Design

Programming enters into the architectural discourse as a novel approach previously not found in known prior art. William Pena and Steven Parshall's book *Problem Seeking* as the first systematic study on the issue of programming in architecture sets the basics of conceptual and theoretical agenda in a context. The book aims to be "a living document of tools, techniques and guidelines for the future advanced programmer" *(Pena & Parshall, 2001, 9)*. Although it has been several times revised and reissued by its authors its main structure has remained the same. In the book, problem seeking *(programming)* is structured according to the five levels in the act of programming. They are described under five key terms of goals, facts, concepts, needs, and the problem *(Pena & Parshall, 2001, 12)*.

Parallel to the work of Pena and Parshall, from the early 1970s to the present, Henry Sanoff played a key role in developing the conceptual and methodological content of the discourse and strengthened the belief in programming as the core operational component for systematizing the design process. In the foreword of *Methods of Architectural Programming (1977)* Richard Dober argues that "programming is a necessary activity in any design process that claims to be responsive to user needs"; it "brings maturity to the architectural arts" *(Sanoff, 1977, vii)*. As opposed to the early modern movement's star architects, programming does not include any reference to even a single designer as architect. Sanoff emphasizes "prescriptive" nature of the program as a "communicable statement of intent" on both the design process and the designer. One of the primary interests of programming studies is to make design socially valid instead of personally. Therefore, terms program and programming refers the whole idea of design settled around the issues of "participation" and "user needs" *(Sanoff, 1977, 4)*.

Another basic reference is Donna Duerk's book *Architectural Programming: Information Management to Design (1993.)* In the introductory part of the book, she points out that programming begins to show itself in architectural publications as a result of the disappointment of *Pruitt Igoe Housing* example, which is designed by Minoru Yamasaki at the end of 1960s.

1 *Design Methods movement refers to "a response to a world-wide dissatisfaction with traditional procedures" and a critical reexamination of methodological problems in design at the second half of the twentieth century. The 'Conference on Design Methods', which was held in London in 1962 is regarded as the launch of design methodology as a field of enquiry aims at objective problem posing and solving in design and application of novel scientific methods (borrowed from operational research and management), to the problems of the post-World War II period. The movement advocates systematic methods of problem solving by focusing especially on decision-making procedures in design.*

Duerk explicates that in the USA, 1970s represent the decision of demolishing the building blocks due to poor social and behavioral quality, and to their becoming crime scenes. The situation has an influence on the determination of programming as a standard service by the *American Institute of Architects (AIA).* According to Duerk, coming after the *Pruitt Igoe* phenomenon, 1980s are the years that programming courses are embraced and willingly given by architectural schools *(Duerk, 1993, 1)*. This new programming atmosphere, which she underlines, also involves an appreciation and admiration to the roots and values of the functionalist architecture *(Duerk, 1993, 1-2)*.

Duerk emphasizes that the field of programming must be in charge of building the "pragmatic foundation of design information for each design project". In this way, programming is defined as a study area, which is supplied with a pragmatic point of view in structuring functional basis of design *(Duerk, 1993, 3)*.

Duerk's introduction puts forward the fact that there are two kinds of approaches to modern architecture within the programming discourse in 1970s and 1980s. The one is based on the assumption of modern architecture as a mere stylistic experimentation, and the other is the assumption of modern architecture as a rationalistic experimentation. While the first group severely criticizes the outcomes of the modern period, the second group, including Duerk, no matter how weak or naïve the modern functionalist project was, appreciates it as a methodological approach and bears on the idea that programming is part of the functionalist lineage.

As in other programming sources published between the years from the 1970s to 1990s, Robert Kumlin's book *Architectural Programming (1995)* examines programming as the foundation of the design process. Differring from the rest, it points out the emerging theoretical framework of programming that entails programming as a dynamic and evolving phenomenon. Yet, on the main assumptions of programming, he shares the similar perspective with the previous researchers.

On the other hand, in the book, he also remarks the importance of "re-structuring the entire program at the highest level of abstraction to achieve the true mission of the facility". He adds, "programming at this level is a very creative and exciting process" *(Kumlin, 1995, 192)*. With this position, he challenges the common assumption that programming does not address creativity. Under such condition, creativity becomes a responsibility shared by both analysis and synthesis. This perspective is anchored on the notion that programming and designers are closely linked.

Edith Cherry, as being another basic reference to programming, focuses on the epistemological values of programming. She argues that architectural programming is inherent in our conception of the world, and it "has always occurred at some level of consciousness". To exemplify her claim, she points out some scenarios:

1) To finding a shelter for not to be wet under the rain, *(unconscious behavior of problem solution)*;

2) To adding a new room to a house for a newborn baby according to the traditional local building techniques, *(conscious behavior of problem solving, but unconscious design act)*; and

3) To decide the need of a new school building and consciously prepare a program to build it step by step by various service providers and users *(school board, architectural programmer, designer, contractor, students and teachers…etc.) (conscious effort to design) (Cherry, 1999, 4-5)*.

For Cherry, each scenario illustrates a different approach to solving problems of shelter need. The main difference among them is the degree of consciousness for the people who are involved in the design process *(Cherry, 1999, 5)*. In that sense, she assumes a closer, more conscious,

and a more interactive relation between programming and design. Her position is important in the sense that such relation might be a sign of the possibility of a designerly framework.

Late 1990s saw the emergence of theorization and critical perspectives on architectural programming. As stated by Robert Hershberger architectural programming differs from others in three points: in its being educational, discursive, and especially in its emphasis on "qualitative, or value issues" *(Hershberger, 1999, p. X)*. Similar to previous researchers, he defines programming as a definitional stage of design *(Hershberger, 1999, 1)*. Yet, his position differs on the relation of architectural design. In his book, programming is expressed through the motto of Calvin C. Straub: "The program is the design!" *(Hershberger, 1999, 3)*.

Based on that framework, Hershberger's *Architectural Programming* focuses on the issue of architectural form. He does not exclude, ignore or despise architectural form as commonly done by programmers. On the contrary, he underlines that qualitative issues and especially form are the most vulnerable parts of architectural programming. His hypothesis is that architectural form "is not simply the result of physical forces or any single causal factor, but is the consequence of a whole range of socio-cultural factors seen in their broadest terms" *(Hershberger, 1999, X)*.

Although Hershberger points out a dialogue between programming and design, his critique does not directly target its established methodology: analysis-synthesis. On the other hand, compared to the mainstream programming discourse, Hershberger has rather a holistic approach, which combines design and programming.

Short history of studies on architectural programming summarized on the above shows that, first wave of programming discourse focuses on a clear understanding of a new technical professional field called programming[2]. On the other hand, second wave *(which is essentially regarded as the critical step in the evolution of a computational approach on programming)* appears to be more a result of computerization.

2.3 Programming & Computation

Since the 1960s, design has been studied from the point of view of information systems and knowledge-based approaches. In the 1970s and especially in the 1980s, "formalization, representation and manipulation of knowledge in computers have made it possible to construct knowledge-based design systems" *(Coyne, Rosenman, Radford, Balachandrian, & Gero, 1990)*. Success of Allen Newell and Herbert Simon on the issue in the 1970s show that the new computationalist[3] agenda has been naturally internalized by behaviorist architecture as part of ongoing programmatic tradition. Despite their short history, computers and their associated subject areas are believed to have the potential to produce fundamental changes in design.

There are numbers of reasons why the analysis-synthesis model proceeded to flourish as a part of the computationalist agenda of programming. Firstly, at the beginning of the 1960s, computers had become more practical and personal as opposed to their earlier forms –i.e. desk calculators, punched-card equipments, and analog computers. Since the programming discourse was developed on the potential of these earlier tools, programming praxis has simply kept pace with their advancement. Secondly, computers had reached enormous storage and computation capacity and started to implement processing of exceedingly complex data, which are far too complex for human designers to process. Such a capacity was in a way seen as a promise in contribution to a solid progressive development within the field of design by means of programming.

Design research as the title of a new "systematic search and acquisition of knowledge related to design and design activity" *(Bayazit, 2004, p. 16)* established the grounds for the foundation of design computer relation in the early 1960s. Pioneering studies

2 *In her "Building Programming: From 'Problem Seeking' to Architectural Values", Pınar Dinç reviews the phenomena of "building programming" within the field of architecture between the years of 1977 and 1999 by examining the pioneering books of the period. In her study, she points out a change of focus in understanding the term over time from "having a practice based narrow area" to "a broad spectrum that covers the whole constituents of the field of architecture." For a detailed information see (Dinç, 2002).*

3 *In the design research discourse, the term "computationalist" was first used by Liddament in 1999 for expressing one of the most pervasive and influential paradigms operating in the contemporary design research. It refers incorporating computation as part of the problem solving process. The term also reflects a critical perspective toward the positivist epistemology (Liddament 1999, 55-56).*

of Gregory, Ward, Broadbent, Asimow, Alexander, Archer, Jones, Gordon and Osborn as well as conferences held by the *Design Research Society (DRS)* in London *(1962)*, Birmingham *(1965)*, and Portsmouth *(1967)* shaped and determined the main problems and principles of the new framework *(Cross, 2007)*. In addition to conferences and books, journals have also played a crucial role and encouraged development of a computationalist perspective in 80s.

Since research on architectural program and computation has been mostly conducted from a variety of partial perspectives at the background of the main discussion of the design research, relation of the two has evolved rather implicitly. As an exception to this pattern and as one of the pioneering examples in associating the mathematical, computational logic with programming in the area, Christopher Alexander's *Notes on the Synthesis of Form (1964)* focuses on a programmatic conception of architectural design in detail. The study refers the concept "design" as a "well-formulated" goal and underlines the primary goal of the researcher as evolving a "language of design" based on the process rather than the product. In his book, Alexander emphasizes the role of the program in design as fundamental. By following inductive logic, he argues that form is the result of forces in the environment and "physical clarity cannot be achieved in a form until there is first some programmatic clarity in the designer's mind and actions" *(Alexander, 1973, p. 15)*. With the help of set theory, he pursues an objective to explore symbolic conceptualization of the design problem as built out of mathematical entities. In such a framework, the goal of program is to define design problem as "decomposition" - that is, proposing "precisely definable operations" via structuring "a hierarchical nesting of sets within sets". Program in this context is then "a reorganization of the way the designer thinks about a problem" *(Alexander, 1973, pp. 81-83)*. Fitted well with the inductivist logic, such perspective served as the locomotive of a program-biased computational understanding of design.

In the 1960s computer was regarded as the revolutionary element *(instrument)* of design. Yet, unlike the optimism of the 1960s, in the second half of the seventies *(at least in the academic domain)* positivist programmatic trend has slowed down following self-critical reflections on the previous "first-generation"[4] methods. Design research journals, although relatively late as compared to the other media, were mainly launched in the same period when critiques of induction and program-biased design were growing rapidly.

As part of such a context, and being one of the oldest in the field, the *Design Studies* journal has been launched in 1979[5] to serve a change from a purely positivist orientation of the previous phase of design research to a new, rather unknown post-positivist phase. In consistent with such refreshing atmosphere, the journal was declared unique and pioneering in approaching the design research field *(Cross, 2007, p. 3)*. By referring Toynbee's "new country" in the editorial part of the first issue, Sydney Gregory pointed to a change of paradigm from *Design Methods* to a hopeful "uncertainty" *(Gregory, 1979, p. 2)*. Yet it is seen that within the course of its publication, especially on the issue of computation and program, the journal does not exactly follow this initial guideline. It both embraces criticisms toward the program-biased approaches *(such as: (Darke, 1979) (Archer, 1979) (Steadman, 1979) (Fowles, 1979)*) and at the same time continue to encompass inductivist positions of computer involvement in design as part of its agenda. In the *Design Studies* journal, and in general in the design research, what one will see when one trace the relationship between computer and architectural program then is two counter-position but mainly a good number of positivist computationalist articles.

Between the mid-1979 and 1980, linking computers to design by means of program-

4 *The term "first-generation" connotes a library of methodologies, which guides the early years of the Design Methods movement. It is based on the idea that the design process (like any other industrial production process) should be entirely explicable like machines. The main motivation of the model is the idea of externalizing the design process (glass box approach). Its intention is to resolve the "conflict that exists between logical analysis and creative thought", and to keep "design requirements and solutions completely separate from each other" (Jones, 1963). In this model, the logical and the creative parts are assumed to be reunited by the idea of "finding" the solution within the synthesis part. However, contrary to the hopes of many, due to the poor representation of actual design activity, the model was later declared to have failed by those who came to be the "second-generation".*

5 *The Design Studies journal was launched as part of the foundational objectives of the Design Research Society (DRS) at the end of a period of a joint journal project titled Design Research and Methods (DMG-DRS) and initiated in collaboration with the Design Methods Group (DMG) in the 1970s. As one of the first international institutionalized journals in the field, Design Studies established the foundations of the design research. As such, together with other pioneering media, it provides a representative view of design research and allows us to study program in parallel to the development of design research starting from its early days. From the late 1980s until today widening of the design research and ease of accessibility of the information has led to an increase in the amount of sources, whereas, because their length of life prevent them from contributing to the preliminary and foundational discussions, on the issue of the evolution of the concept program, such material will be referred to as incomplete.*

ming clearly is one of the main missions of *Design Studies*. By looking at the number of such articles in the very first issues, it can be easily said that it demands a closer designer-computer relationship. For example, John and Carroll's article, "The Psychological Study of Design", is a study that is structured with the intention of examining design in the field of computer science. In a similar fashion, in 1980, in the fifth issue, Sydney Gregory reviews Yourdan's book, *Managing the Structured Techniques.* He points out the importance of advancing programming methods in design via computation, and identifies "structured programming" as having quite a potential for improving the field *(Gregory, 1980, 316)*.

Alwyn Jones in a book review at the third issue in 1980 introduces to readers computer programming as a new tool and a component of "system analysis" to infiltrate participation in design. Jones claims that system analysis has potential to contribute to design and it "is in fact truly a designer". It deals with the problem of "how to reach creative types within the framework of known procedures?". It is then "a good-old-fashioned disciplined approach to work, with new logical techniques relevant to the science of computer programming" *(Jones, 1980, 180)*.

Charles Eastman's article "Information and Databases in Design", in the same issue, defines the link between design and computer from an information processing perspective. Eastman claims that "designing can be studied as an information processing task", and argues that the transfer of information storage, management, and processing techniques, have a great value to develop "manual" design *(Eastman, 1980, 146)*.

In the same year in the journal, Berger published an article titled "Artificial Intelligence and its Impact on Computer-Aided Design", in which he lists the advantages and disadvantages of artificial intelligence *(AI)* in design. In the next issue of the same year, Gero's article "Computer-aided Design by Optimization in Architecture", re-examines design and analyzes how computer-aided optimization is relevant to design. In another article, based on the idea that "any problem that can be defined as computationally, can be solved by optimization", Gero, tries to re-define decision-making procedures in design from the perspective of programming *(Gero, 1980, 227)*. Cooley's article in the fourth issue claims the importance of the contribution of computers to design and yet, warns designers of the possibility of their becoming *Trojan* horses, which invite *Taylorism* to design, and damage the creativity grounded on human-centered *tacit* knowledge.

At this point, it is important to note that, up to the 1980s developmental progress in computer-aided design studies have matched with the progress and advanced institutionalization of the design research community. After the second half of the 1980s however, diversity of distinct positions in the area becomes noticeable. Framed in this context, two positions have appeared. The first one as upheld especially by *The Design History Society* focuses on the post-positivist issues via foregrounding traditional tools of art and architectural theory. The position is best exemplified with its journal the *Design Issues (1984)*. After the *Design Studies* journal, *Design Issues* broadened the boundaries of inquiry into a neglected area: history theory and criticism. From the revolutionary tones of pioneering books and conferences launched in the 1960s, *Design Research Society* has largely came to defend a position where researchers set themselves off from "over-simplified pragmatism" of the previous decade. As part of this atmosphere, in the editorial part of the first issue, *Design Issues* declared its commitment to solve the problem of poverty arises from the previous reductionist perspectives nested in design tradition and education in the United States. It also declares vitality of reengagement with "history theory and criticism" in the sense of understanding

design as "a significant social and cultural practice" *(Margolin, 1984, p. 3)*. The journal takes into consideration areas that were once repressed *(i.e. history and culture)* by asking questions such as "To what extent can history contribute to the understanding of what design is and what the designer does?" *(Dilnot, 1984, p. 5)*. Post-positivist background of the journal provides researchers to re-engage previous tools and traditions, but it would not be correct to say that this contribution results in a change in the concept program. The second position on the other hand, is upheld by an extremely specialized younger study field: "digital design". As best exemplified by studies of Rivka Oxman, theorization of digital design is a notable attempt in the last decade in describing a relationship between design and programming.

The term "digital design" refers to a unique phenomenon, an advanced computational position, which radically challenges traditional means of designing/programing in architecture. It is a methodologically unique form of design "a new set of technologies and unique media of design that are transforming our traditional definitions and concepts of design" *(Oxman, 2006, p. 238)*. In that sense, it is also an experimental research project, an effort to test and improve our understanding of a digitally based design process.

In digital design, the issue of program has been shaped by the techno-utopian revolutionary essence embodied by the evolving digital design theory. In such a framework, program is based on a demand for a "cultural transformation of root design concepts" such as normative, static, and typological aspects by offering alternative proposals *(Oxman, 2006, pp. 232-233)*. As part of such perspective, digital design is described as a complex organizational system at the high end of digitally networked environment. Hence, the question of what the relation between digital design and program is largely related to the network and the role of the designer.

As digital design media become more complex and more demanding with respect to knowledge of multiple types of software, knowledge of scripting languages, and the manipulation and maintenance of complex data models, a new generation of digital design specialists is emerging... The thought of the designer as digital toolmaker reflects both the potential for customizing digital design media as it does the necessity for specialist knowledge needed to operate such media. So presently, the idea of a class of 'digerati', or digital literati as advanced digital systems designers appears to be an accurate description of the contemporary situation *(Oxman, 2006, p. 262)*.

In a system such as this, the role of the designer is described as "digital design specialist", and "digital toolmaker" which ensure a high level of specialized knowledge to operate digital media *(Oxman, 2006, p. 262)*. As identified at the center of every process in design, and by managing, controlling, and manipulating it, the digital systems designer, or as coined by Oxman "digerati", reminds of a sophisticated version of the programmer of the 1970s. As explicated by Mark Burry in detail *(Burry, 2011)*, it is conceivable that having the knowledge and power of scripting languages, the digital designer is an upgraded version of a programmer.

Scripting is a rather loose term by any definition and in this primer can be taken to mean computer programming at several levels. For the novice dabbling at the more accessible end of the user spectrum, scripting is the capability offered by almost all design software packages that allows the user to adapt, customize or completely reconfigure software around their own predilections and modes of working. At its most demanding for the emerging connoisseur, scripting can refer to higher-level computer programming where, in the 'open-source' environment, 'libraries' of functions can be combined with preconfigured routines *(algorithms)* as a means to produce manufacturer-independent digital design capability *(Burry, 2011, p. 8)*.

Yet, different from the expert programmers of 1970s, digital design has taken the limits of programming a step further. For Oxman, digerati has two duties: "scripting" and "manipulation and maintenance of complex data models" *(Oxman, 2006, p. 262)*. For her, while scripting is a technical skill, a specialty, as in fine use of a tool, manipulation and maintenance of complex data models is where the creativity and design comes. On the other hand, for Burry, the term "scripting" seems to involve both:

> *'scripting language' is often synonymous with 'programming language': it is the means by which the user gives highly specific instructions to the computer with which they are interacting. At a semantic level, it is possible that the designer is less likely to flinch at the term scripting than they might at the term programming, for it is quite clear that most of the designers who use computers as a core part of their digital practice do not automatically turn to programming to form part of their repertoire. By not doing so users at once place their entire trust in the software engineers in the expectation that those anonymous collaborators have thought through all that might be wanted by the designers, just as they are conceding that what seems on occasion endless manual repetition is an acceptable use of their time when they could otherwise have been seeking some degree of automation. Software modified by the designer through scripting, however, provides a range of possibilities for creative speculation that is simply not possible using the software only as the manufacturers intended it to be used. Because scripting is effectively a computing program overlay, the tool user (designer) becomes the new toolmaker (software engineer) (Burry, 2011, p. 9).*

Idea of digital design that associates revolution and "systematic", "scientific" utilization of advanced technology is well consistent with a typical analysis-synthesis model. As part of the design research tradition, whose roots still strictly anchored to the inductivist principles, digital design describes and despises traditional design methods as "stylistic" and prioritizes "process" over "product" *(form)*. It argues that in the digital age, "change in the professional culture of architecture" gives rise designers to "transcend stylistic agenda" *(Oxman, 2008, p. 100)*. Yet, although it adopts a similar approach with its precedents, digital design distinguishes itself from the typical form of analysis-synthesis. As such, it claims to modify conventional "analysis-synthesis-evaluation" scheme by expanding it toward a "performative organizational systematic process" *(Oxman, 2008, pp. 107-108)*. The resultant change redefines program as a continuous performative programming activity continued throughout the design process and modify synthesis with generation. Despite the obvious emphasis on the newness and uniqueness, hidden behind this framework there still lies a positivism and a danger of determinism. As in the classic inductivist phrase "form follows function", "the actual form emerges from a process seeking for optimal performance" *(Oxman, 2008, p. 107)* is no different.

As having strong influences on the evolution of the concept program, tradition of computation in design results in two main outcomes. The first one is that both handling much varied and complex information and utilizing it towards a finished product opens up a new unknown research area in design. The second one is that such a situation obliges the role of a traditional architect to change *(Broadbent, 1979) (Coyne, 1995, 31) (Oxman, 2006) (Oxman, 2008)*. Computer aided advancement of analysis-synthesis seem to result in architecture's profound alienation from its own past. Since, such framework tends to describe design as a pure process of induction; it demands from designers a clear-cut rejection of the so-called "traditional" design methods and of the architectural design culture.

In parallel to these ends, positions regarding the role of computer in design fall into two camps. While for the first camp computer is taken as an advanced tool in handling design problems described and controlled by human designers *(i.e. Cooley)*, for the second camp it is taken as an immature but powerful source of knowledge, a potential new research area on the way to challenge the whole design culture *(i.e. Oxman)*. Hoping to reach a "truly creative

use of the computer" and a revolutionary description of design, this second position advocates exploring the "black box of programming" as the main focus of digital computation *(Terzidis, 2006, p. vii)*.

With such a prophecy of revolution, position of computer in the field of design tends to propound a firm critical attitude toward architectural history and culture advocated by program-biased computationalist approaches and a shift and displacement of positions *(a paradigm shift)* in understanding the field from the evolutionary to the revolutionary. As emphasized by Terzidis, while the goal of the CAD is to "free the designers from repetitive, time consuming tasks" it is also having the aim of opening up fresh, new, unusual design experiences by endowing the designers alternative, innovative, "frame braking" pathways *(2006, pp. 53-54)*. For him, due to the unique "external" explorative nature of digital computation, results of integration of computer *(inductive logic)* to design might be expected to associate with Kuhnian concept of the "paradigm shift." A paradigm shift is: a gradual change in the collective way of thinking. It is the change of basic assumptions, values, goals, beliefs, expectations, theories, and knowledge. It is about transformation, transcendence, advancement, evolution, and transition. While paradigm shift is closely related to scientific advancements, its true effect is in the collective realization that a new theory or model requires understanding traditional concepts in new ways, rejecting old assumptions, and replacing them with new *(Terzidis, 2006, p. 59)*.

In the digital design discourse, such an argument has been used in the comparison of the humanistic and non-humanistic design approaches. As argued by Terzidis, due to the distinctive capacity of digital computation, the human understanding extends its limits towards unknown fields. As such, computer provides designers a possibility to reach an external and foreign perspective through which a paradigm shift might be happen. On the other hand, the concern is that issues of computation in design seem to be stuck largely in "humanistic philosophical theories of 60s and 70s" *(Terzidis, 2006, p. 55)*. Yet, despite the power of such theories, to grasp the "hidden mechanisms" of computation one should focus on "actual implementation *(i.e. programming)*" rather than on the humanistic problem of the relationship set between the human mind and the computer. In accordance with this, since not the users *(or spectators)* of tools but inventors of tools have set the workplace, the long been expected paradigm shift occurred "not in the designer's mind but in the programmer's mind" *(Terzidis, 2006, p. 54)*.

From the late 1960s to 1970s, with an emphasis on the pragmatics of the market and experimental teaching, "the study of design methods tended to give way to the study of the principles for erecting and manipulating models of the things or systems being designed" *(Archer, 1981, 31)*. Late 1970s are on the other hand; set the scope of a new degree of understanding based on the academic studies those which intellectually supported by the newly emerged postgraduate design programs. It is the same period that design research "became heavily involved in the development of computer aids to designing" *(Archer, 1981, 31)*. Two decade of study embodied an intense background in rationalization of design, which created a reverence for program and study on methods of complex programming. The pursuit for a systematic understanding and computer involvement in design theory and practice in the late 1970s reshapes the main framework of design research and, therefore, program. With this advanced stage of understanding, previous structure of design research, which was built onto the *Design Methods* ideals *(primarily the aim of handling complexities in a truly systematic approach – which was mainly based on operations research and systems approach)*, this time denotes integration of academic, scientific and computationalist participation. As such, programmatic content in the design research provides us continuous re-assembling of concepts by the leading design research media, which is part of a wider trans-disciplinary

meta-framework. Finally, digital design, as presenting the representative view of the last step of the computational advancement in the 2000s, reviews the concept program provided by the design research and then modifies it toward a "performative" organizational element. Despite all these developments, it is hard to claim that such advancement created an entirely new epistemological exploration of a programmatic understanding, a revolution. On the contrary, it seems that in the 2000s as in the 1960s, on the concept program, design research discourse *(at least in computational studies)* is still predominantly following accustomed patterns of induction.

Although the idea of program and programmability of design processes *(in the sense of describability and clarification)* shape the general trend of the design research tradition, late 1960s and early 1970s gave birth to its counter alternative argument especially in architecture. Following the post-positivist, and especially Popperian epistemology, in such framework the architectural idea is prioritized as the guiding element, or schema, of design process. As such, after relinquishing the authority of decision-making, architectural program has been redefined as a passive agent, which does not imply or point to a certain solution, and does not demand a total control of the design process as opposed to its inductivist counterpart.

3. The Birth of the Counter-Paradigm: Architectural Program from the Perspective of Conjecture-Refutation

Starting from 1950s both pros and cons of functionalism in architecture converged on the fact that the positivist epistemology behind the functionalist project failed. Since then, a post-positivistic Popperian framework emerged in design research and especially in architectural theory. It affected the design discourse by focusing on the following arguments that, design is incompatible with the model described by the *scientific method*, that design is not teleological and deterministic, and that, design does not point to a process which is described as an unbreakable chain of cause and effect as modeled in well-formed problems in the early 60s "design science" campaign advocated in the *Design Methods* movement. From the Popperian perspective, design is rather to deal with uncertain, "ill-defined" problematic situations whose solutions are "implicit in the artistic, intuitive processes" which lead designers to "situations of uncertainty, instability, uniqueness and value conflict" *(Schön, 1991, 49)*.

The changing of epistemologies in design research resulted in both a novel conception of science and an alternative design model. As a consequence, despite the apparent "scientific" and programmatic bias associated with the *Design Method* movement in the 60s, 70s were the years of critical skepticism and awakening period, and in parallel to Popperian epistemology, 80s were the active contribution period for proposing a more coherent model of design accompanied by an alternative conception of program.

The counter-paradigmatic line of evolution on the concept of programming in architecture is important in the sense that it provides foundation for current research issues and agendas. For example, although theoretical groundwork and first implementation was laid many years ago, in the early 2000s there has been a recent revival of interest in inductivism and hence programmatic conception of design. The idea of search for a program-based architecture that could assist architects in their quest for the logical explanation of the design decision processes has been strongly influenced by the general euphoria associated with computerized design and the super-analytical nature of computers. As architectural design became more and more computerized, it has been reduced to a matter of a process whose structure contains: analysis, synthesis, and evaluation, or shortly, analysis-synthesis.

3.1 Epistemological Background of Conjecture-Refutation

In *The Logic of Scientific Discovery,* Popper warns the reader against the

existence of "myth of scientific method" instilled in the explanation of Baconian scientific inquiry. He argues that on the basis of the problem there is the Baconian conception of scientific method which argues that scientific inquiry "starts from observation and experiment and then proceeds to theories" *(Popper, 2005, 279)*.

The Popperian view is basically grounded on the idea of the falsifiable nature of scientific knowledge. It starts with the hypothesis that if we do not know, "we can only guess. And our guesses are guided by the unscientific, the metaphysical *(though biologically explicable)* faith in laws, in regularities which we can uncover - discover" *(Popper, 2005, p. 278)* and "if observation shows that the predicted effect is definitely absent, then the theory is simply refuted" *(Popper, 1957, I)*. As such, in conjecture-refutation, the success of science does not depend on rules of induction, but "luck, ingenuity, and the purely deductive rules of critical argument" *(Popper, 1957, VIII)*.

By following Greg Bamford's words, conjectural critique can be summarized in three points: First, "The idea that scientific inquiry begins with observations or facts is false". Because it is an attempt for an "explanation about what we do not understand", scientific inquiry begins with problems. Second, "there is no logic or method of discovery that will conduct us, and certainly not in the orderly fashion". Unlike the Baconian view, "scientific theories are imaginative constructions which go well beyond whatever they were designed to explain". Third, unlike the analysis-synthesis, conjecture-analysis includes error. Thus, "criticism, or flushing out error is the engine" of the conjectural understanding of the science *(Bamford, 2002, 249-250)*.

In light of these three points conjectural understanding of program is shaped by following arguments. A well-known inductivist *(program-based)* assumption of "design starts from facts" is a myth. The actual procedure of science and therefore design operates with conjectures and conjectures are not derived from factual understanding of the world. Induction makes design decisions only probable rather than certain. Yet, designing *(and especially starting to design)* requires firm decisions. Such decisions do not come from probabilities derived from observation statements *(analysis)*, they derive from unjustified anticipations, guesses, or tentative solutions to problems.

3.2 Conjectural Programming as an Alternative to Analysis-Synthesis

The approach has been advocated and experienced since late sixties by a handful of researchers. As became visible in 1970s, pioneers of the movement demanded a complete disengagement from the Baconian paradigm. Unlike the strong belief in analysis-synthesis inherent in the *Design Methods* movement, and then in the computational agendas, researchers who has been working this area described the scientific method and the design process as incommensurable. For them, exercising Popperian paradigm opens up a promising area for new interpretations to untouched problems in the field such as: creativity, uncertainty, subjectivity, and relation with past design knowledge.

In general, the initiation of the post-positivist canon and especially the conjectural understanding of science within design research was referenced to Broadbent *(1969)*, who is the founder of the title "third-generation" and harbinger of the new conjectural approach, as well as to the notion of "Knowledge and Design" by Hillier, Musgrove and O'Sullivan *(1972)*, who are the pioneers of explicating the possible conjectural methodology. Yet, the epistemological roots of the position comes from a rather disengaged group of Anglo-American academics who have strong ties with the philosophy of science as well as modern art and architectural history and theory.

In the spring of 1963, Royston Landau organized a symposium at the *Architectural Association School* in London on the subject of "the context for decision making

in the arts and sciences". Symposium papers were published in the AA Journal in 1965. The event brought together young academics from both the UK and the USA.

As opposed to the inductivist, positivistic perspective dominant in those years, the group preferred to discuss the issue of technology in the context of architecture and questioned the effects of rapidly increasing technologies -such as computers- on architecture. They clarified the role of technologies in decision-making procedures of design and focused on epistemological consequences rather than methods. In the symposium decision-making was discussed under four major headings, these are: art history and theory *(Ernst Gombrich)*; mathematic, logic and computation *(Jack Cowan)*; history, philosopy and epistemology of science *(William Bartley)*; history, philosophy and epistemology of architecture *(Stanford Anderson, and Royston Landau)*.

The symposium as such frames an earlier effort based upon the post-positivist challenges to positivist rooted design research. It occupies a unique position between two categories of decision-making. One category is distilled from the historical cultural context of art and architecture led by traditions *(paradigms)* which are nearly completely avoided by the dominant design research literature. The second category is distilled from rather abstract mathematical inductivist decision-making procedures *(programs)*. The symposium is important and should be foregrounded not just for its historical uniqueness but also for its potential of stepping over epistemological and positivistic obstacles indoctrinated in the design research tradition.

The new paradigm came to prominence in architecture at a time when design research was often threated skeptically and for a long while, modernist architecture has been the object of popular antipathy. Following Robert Fowles's words, in the 70s we see, "the hard-edged, objective, rational, quantitative and systematic form of design methods has undergone a variety of transformations to become variously accommodated in a variety of forms and in a variety of contexts" *(Fowles, 1979, 16)*. In this period, the worldview of analysis-synthesis was reviewed, evaluated and then labeled simply as unsatisfactory.

In 1972, a report for the state of the art of design research was prepared. In it, Donald Grant points out that, starting with the *Portsmouth Symposium* held in 1967, there is an emerging feeling that "architecture should begin to develop techniques and approaches unique to their own problems and to depend less on techniques borrowed from other related fields like operations research and engineering design" *(Grant, 1972)*.

As part of such evaluations, in 1979, Broadbent announced Popper's conjectures and refutations as the new "third-generation" methods. He severely criticized the idea of design as inductive reasoning *(as defined mainly by participation, collaboration and user issues)*. He blamed the approach for excessively involved in scientistic methods and claimed that non-participatory methods in design "have been remarkably successful", on the other hand, participatory ones are incompatible with the phenomenological approach. He also argued that collectivist approach was never intended to involve the architect in the design process, but to exclude him/her. Participation in planning is an ideological tenet of left-wing political dogma founded upon "that 19th century version of Utopia in which all men being equal lived together in collaborative harmony" *(Broadbent, 1981)*.

Broadbent simply reacts to the previous era of design research and accuses its positivist agenda as being reductionist based on the shortsighted description of all aspects of design under the terms participation and/or collaboration. Since in participatory/collaborative approach, the decision-making process falls under the control of "utility", cultural issues are not considered. For him, this is a failure even beyond the point reached by the model

proposed by Sullivan at the beginning of the last century. Reductionist formulation of design as inductive reasoning exemplified by first-generation design methods movement and then second-generation participation/collaboration strategies mainly fall short emphasizing and conceptualizing the place and importance of creativity in the making of form.

Beginning with an introductory article by Jane Darke written in 1979, the leading critiques were followed by a special issue of the journal of the *Design Studies*. Based on a research conducted under the editorship of Purcell and under the governance of Schön *(MIT)*, July issue of the journal in 1984 has a particular importance in the sense that it provides new initiatives not only for design research but also for programming. The issue brings an international research atmosphere to the journal, which it relatively lacked before. It also incorporates areas, which the tradition of design research tends to exclude, such as history/theory and philosophy. However, the most striking of all, for the first time in the history of the journal, a group of articles *(including Schön, Anderson, Andreotti, and Metallinou)* have been grounded their arguments on a phenomenological framework, rather than on scientific method.

Article of Donald Schön *(1984)*, in the issue represents a significant alternative perspective in the field due to its tendency to re-engagement with the master-apprentice structure, and with the architectural tradition *(as natural components of design)*, which was traditionally rejected by design research and by the approaches, which are based on programming. The idea of program in Anderson, Andreotti, and Metallinou's studies is guided by an advanced Popperian perspective. Their works are also unique in the field and the journal since they define architectural program as a series of research programmes. As guided by a Lakatoshian *hard core,* here hypothesis/program is assumed to be surrounded by a *protective belt* of *auxiliary hypotheses*/programs, which vary in response to unfolding discoveries and problems throughout the design process or a series of processes.

As opposed to the a-historical, a-cultural dominant paradigm of the *(naïve)* scientific method, and its main proposals that "architectural design should be changed all over!" "it should become science!". Anderson brings to the fore Lakatos and presents a post-positivistic counter argument in the context of architecture. For the Lakatosian perspective, since both architecture and science are "constructions of culture", they are not alien to each other. In both fields", research program is built around a particular problem situation" which means that both are derived strongly from "temporal and historical" context *(Anderson, 1984, 148)*.

It is with this shift of understanding that we saw the emergence of a new critical conjectural contribution to design research. In the journal, Schön adapts Lakatos' ideas of a research program to the field of design as follows:

Architectural designing can be understood as a kind of experimentation... Making a design move in a situation can serve, at once, to test a hypothesis, explore phenomena, and affirm or negate the move... The very invention of a move or hypothesis depends on a normative framing of the situation, a setting of some problems to be solved. In the evaluation of a move, the designer asks whether he gets what he intends and whether, on the whole, he likes what he gets. When moves function in an exploratory way, the designer allows the situation to 'talk back' to him, causing him to see things in a new way - to construct new meanings and intentions. It is only within the framework of an appreciative system -- with its likings, preferences, values, norms, and meanings -- that design experimentation can achieve a kind of objectivity. Although a designer's likes and dislikes are subjective, and may even be arbitrary, he can discover, independent of mere think-so, whether his moves have produced something he likes (Schön, 1984,

132).

Starting with a hypothesis, testing it, exploring phenomena, and affirming or negating to move… these steps are the main structure of the paradigm of conjecture-refutation. However, the most significant contribution of Lakatos comes from his remarks on the critical conventionalism involved in the nature of programme. By following Lakatos, Anderson states that, the "conventional element of science has invaded the very core of the scientific enterprise. Convention is an aspect of that which assures the maintenance of the programme" *(Anderson, 1984, 148)*. This position of being opposed to the positivist roots of design research, being opposed to starting from scratch each time, is not the accustomed way of describing design; it is even against the roots of design research. Presumably it is this very shift in position, which provides a link between architectural knowledge and contemporary inquiries of design as well as link between architectural historians like Stanford Anderson and design research.

Overall, within the conjectural, phenomenological framework of both design decision symposium held in 1963 and the special issue of design research published in 1984, program is viewed as a core that controls the whole. On the other hand, it is also taken as a flexible structure which is fed by a set of auxiliary hypotheses *(protective belt)* surrounding the main design idea at the core. In both cases, program is defined as a decision-making mechanism, which provides observation and clarification of the works of the masters. In this sense, researchers who work on conjectural paradigm do not confine design within a frame of determinist programming but rather they reinterpret it in a new context to emphasize various aspects of creativity.

Conjectural studies focus on final products *(cases, exemplars, precedents, etc.)* and try to describe design *(processes)* through previous knowledge *(previous architectural works)*. In that sense, conjectural approach on design emphasizes that what comes out of past examinations can be used to produce new solutions *(via reprocessing and reinterpretation)*. This leads researches to put emphasis on issues such as evolution, transformation, adaptation, and also reinterpretation and reevaluation of design ideas.[6] Though it is flexible, the design idea is deemed holistic and not conceived through bits and pieces. For this view, this *(preserving and protecting the core even it transforms, bends, and changes throughout the process)* is the essence of good design which should be observed in the works *(and processes)* of experts, and in turn be followed in the design education.

4. Conclusion: Towards a Reconsideration of the Architectural Program

Throughout the evolution of the *Design Methods* movement, first we saw program as a separate analytical pre-design stage, and then, as a meta-database and meta-scheduling mechanism to control the design process from the beginning to the end, and sometimes as a deterministic experimental model, a meta-algorithm for reaching out automated design ideals. However, neither of these solutions fit well to the subject matter of a well-known campaign studied under conjecture-refutation.

Starting from the middle of the twentieth century, post-positivist theories of knowledge have provided a base for the criticism of the foundational Cartesian view of knowledge. For design research, which caught the tail-end of the positivist tradition, the post-positivist third-generation caused a traumatic *(unexpectedly soon)* re-evaluation. Indeed, the crisis in *Design Methods* was triggered by the pride in the factual, analytical thinking and the wish to make induction the main tool for design. For the post-positivist perspective, design as "congeries of conspicuously disparate parts" was led to a quite problematic, unfitting, explanation *(Rowe, 1982/83)*. Yet, the paradox is that despite the diagnosed problems throughout the 1970s, design research is still reluctant to continue with a more fitting epistemology. Instead, it prefers to stay in the inductivist realm

6 *Emphasis on the design ideas motivates conjectural understanding to create an epistemological perspective, a designerly body of thought that pays close attention to the existing artifacts. In that sense, one might see the similar issues as having been studied by Formalism, which is one of the most fertile sources of post-positivist mode of thought. Although there have never been a single unified identity for explaining formalist tradition/s, they in general provide design a rich conceptual library of tools such as opacity, defamiliarization, literariness... etc. For a detailed information on the issue, see: (Anay, 2012).*

and focus on to develop rather restorative methodologies.

The inductivist approach as the stereotypical representative of the scientific method, as the assumption of acquiring knowledge through objective collection of observations, nurtures the field's resistance to the conjectural paradigmatic understanding of design in two ways. First, in design research in general there is an assumption that the paradigmatic turn is to reverse the fundamentals of the project of design research. It is in a way turning back to the past, to the traditional pre-scientific understanding of design and therefore a failure. Second, the techno-utopian and futuristic stances as design research ideals deemed the paradigmatic understanding as an irreconcilable, counter-paradigm.

As a result of these, considering the architectural program, the state of design research today seems to still house two opposite positions. On one side, there is a framework, illustrating design primarily as a collaborative, participatory, interdisciplinary, mostly computationalist programming activity and the designer as a passive translator, on the other side, there is another framework describing design as a primarily conceptual, ideational, cultural phenomenon and the designer as a more active, independent expert guesser, conjecturer, or speculator. First one reads design as the rise of programming, second one normalizes the phenomenon on account of a more paradigmatic understanding.

Bibliography

Alexander, C. 1973. *Notes on the Synthesis of Form*. seventh edition ed. London: Oxford University Press. (First published by Harvard University Press. in 1964)

Anay, H. 2012. (Epistemological) Formalism and its Influence on Architecture: A Concise Review. *A|Z Journal of the ITU Faculty of Architecture*, 9 (1).

Anderson, S. 1984. Architectural Design as a System of Research Programs. *Design Studies*, 5 (3).

Anderson, S. 1984. Architectural Research Programmes in the Work of Le Corbusier. *Design Studies,* 5 (3), pp. 151-158.

Andreotti, L. 1984. Conceptual and Artifactual Research Programmes in Louis Kahn's Design of the Philips Exeter Academy. *Design Studies*, 5 (3).

Archer, B. 1965. *Systematic Method for Designers*. London: The Design Council.

Archer, B. 1970. An Overview of the Structure of the Design Process. In: G. T. Moore, ed. *Emerging Methods in Environmental Design and Planning.* Cambridge, Massachussets, and London, England: The MIT Press, pp. 285-307.

Archer, B. 1979. Design As a Discipline. *Design Studies*, 1 (1).

Archer, B. 1981. A View of The Nature of Design Research. In: *Design:Science:Method.* Guilford: Westbury House, pp. 30-47.

Archer, B. 1999. *On Methods of Research*. Ankara: METU Faculty of Architecture Press.

Bacon, F. 1905. *Novum Organum.* New York, P.F. Collier & Son. (First published in Latin in 1620)

Ball, L., Onarheim, B. & Christensen, B. T. 2010. Design Requirements, Epistemic Uncertainty and Solution Development Strategies in Software Design. *Design Studies,* 31 (6), pp. 567-589.

Bamford, G. 2002. From Analysis/Synthesis to Conjecture/Analysis: A Review of Karl Popper's Influence on Design Methodology in Architecture. *Design Studies*, May, 23 (3), pp. 245-261.

Bartley, W. W. 1965. How Is the House of Science Built? The Growth of Scientific Knowledge. *AA Journal,* pp. 213-218.

Bayazit, N. 2004. Investigating Design: A Review of Forty Years of Design Research. *Design Issues*, 20 (1), pp. 16-29.

Brawne, M. 1995. Research, Design and Popper. *Architectural Research Quarterly*, pp. 10-15.

Broadbent, G. 1969. Design Method in Architecture. In: G. Broadbent & A. Ward, eds. *Design Methods in Architecture.* New York: George Wittenborn Inc., pp. 15-21.

Broadbent, G. 1979. The Development of Design Methods - A Review. In: *Design Methods and Theories*. s. l.: s.n., pp. 41-45.

Broadbent, G. 1981. *The Morality of Designing.* Guilford, Westbury House.

Bunnin, N. & Yu, J. 2004. *The Blackwell Dictionary of Western Philosophy.* s. l.: Blackwell Publishing.

Burry, M. 2011. *Scripting Cultures: Architectural Design and Programming.* Chiscester: John Wiley & Sons Ltd.

Cherry, E. 1999. *Programming for Design: From Theory to Practice*. s.l.: John Wiley & Sons.

Cowan, J. D. 1965. Some Principles Underlying the Mechanization of Thought Processes. *AA Journal*, pp. 251-257.

Coyne, R. 1995. *Designing Information Technology in the Postmodern Age*. Cambridge Mass.: The MIT Press.

Coyne, R. et al. 1990. *Knowledge-Based Design Systems.* s.l.: University of Sydney.

Coyne, R. & Snodgrass, A. 1995. Problem Setting within the Prevalent Methapors of Design. *Design Issues,* 11 (2), pp. 31-61.

Crilly, N. 2010. The Rules that Artefacts Play: Technical, Social and Aesthetic Functions. *Design Studies*, July, 31 (4).

Cross, N. 1979. Responsible Design. *Design Studies*, July, 1 (2), pp. 127-128.

Cross, N. 1994. *Engineering Design Methods*. Second edition ed. Chiscester: John Wiley & Sons.

Cross, N. 2001. Designerly Ways of Knowing: Design Discipline versus Design Science. *Design Issues*, Summer, 17 (3).

Cross, N. 2007. Forty Years of Design Research. *Design Studies*, 28 (1), pp. 1-4.

Darke, J. 1979. The Primary Generator and the Design Process. *Design Studies,* 1 (1), pp. 36-44.

Descartes, R. 1899. *The Method, Meditations, and Selections from the Principles of Descartes.* Edinburgh: Blackwood and Sons.

Dilnot, C. 1984. The State of Design History Part I Mapping the Field. *Design Issues*, 1 (1).

Dinç, P. 2002. Building Programming: From "Problem Seeking" to Architectural Values. *Gazi Üniversitesi Mühendislik Mimarlık Fakültesi Dergisi*, 17 (3), pp. 101-119.

Dorst, K. 2008. Design Research: A Revolution-Waiting-to-Happen. *Design Studies,* 29 (1), pp. 4-11.

Duerk, D. 1993. *Architectural Programming: Information Management for Design.* s.l.: John Wiley & Sons.

Eastman, C. 1980. Information and Databases in Design. *Design Studies,* January, 1 (3).

Feyerabend, P. 1993. *Against Method*. New York: Verso. (First published by New Left Books in 1975)

Fowles, R. A. 1979. Design methods in UK Schools of Architecture. *Design Studies*, July, 1 (1), pp. 15-16.

Friedman, Y. 1975. *Toward a Scientific Architecture*. Cambridge Mass.: The MIT Press.

Gero, J. 1980. Computer-Aided Design by Optimization in Architecture. *Design Studies*, 1 (4).

Gombrich, E. 1965. The Beauty of Old Towns. *AA Journal,* pp. 293-297.

Grant, D. 1972. *Systematic Methods in Environmental Design: An Introductory Bibliography.* Monticello, Illınois: The Council of Planning Librarians, Exchange Bibliography Number 302.

Gregory, S. 1979. Editorial: Design Studies - the New Capability. *Design Studies*, 1 (1).

Gregory, S. 1980. Structured Programming (book review). *Design Studies*, 1(3).

Gross, M. & Fleisher, A. 1984. Design as the Exploration of Constraints. *Design Studies*, 5 (3), pp. 137-138.

Hershberger, R. 1999. *Architectural Programming & Predesign Manager.* New York: McGraw Hill.

Hillier, B., Musgrove, J. & O'Sullivan, P. 1972. *Knowledge and Design.* Stroudsbourg, Dowden, Hutchinson and Ross, pp. 69-83.

Jones, A. 1980. Ironic Principles (book review). *Design Studies*, 1 (3), pp. 179-181.

Jones, C. 1963. A Method of Systematic Design. In: *Conference on Design Methods*. Pergamon Oxford: Pergamon Press Ltd.

Jones, J. C. 1970 . *Design Methods: Seeds of Human Futures*. 2nd 1992 ed. s.l.: John Wiley & Sons.

Kalay, Y. 2004. *Architecture's New Media: Principles, Theories and Methods of Computer-Aided Design.* Cambridge Mass.: The MIT Press.

Kolarevic, B. 2005. *Architecture in the Digital Age: Design and Manufacturing*. New York: Taylor & Francis.

Kostelnick, C. 1989. Process Paradigms in Design and Composition: Affinities and Directions. *College Composition and Communication*, October, 40 (3), pp. 267-281.

Kumlin, R. 1995. *Architectural Programming: Creative Techniques for Design Professionals*. s.l.: McGraw-Hill.

Landau, R. 1965. The Context For Decision Making: 2. *AA Journal*, p. 251.

Landau, R. 1965. Towards a Structure for Architectural Ideas. *AA Journal,* pp. 7-11.

Lawrence, R. 1982. Trends in Architectural Design Methods- The "Liability" of Public Participation. *Design Studies*, 3 (2), pp. 97-103.

Lawrence, R., 1983. Architecture and Behavioural Research: A Critical review. *Design Studies*, April, 4 (2), pp. 76-82.

Liddament, T. 1999. The Computationalist Paradigm in Design Research. *Design Studies*, January, 20 (1).

Liu, Y.-T. 2002. *Defining Digital Architecture: 2001 FEIDA Award*. s.l.: Birkhauser.

Margolin, V. 1984. Editorial. *Design Issues,* 1 (1), p. 3.

Metallinou, V. 1984. Regionalism as an Architectural Research Programme in the Work of Dimitris and Susanna Antonakakis. *Design Studies,* 5 (3).

Oxman, R. 2006. Theory and Design in the First Digital Age. *Design Studies*, Volume 27, pp. 229-265.

Oxman, R. 2008. Digital Architecture as a Challenge for Design Pedagogy: Theory, Knowledge, Models and Medium. *Design Studies,* Volume 29, pp. 99-120.

Pena, W. & Parshall, S. 2001. *Problem Seeking: An Architectural Programming Primer.* 4th ed. New York: John Wiley & Sons. (First published by CBI Publishing Co Inc., in 1977)

Popper, K. 1957. Philosophy of Science: A Personal Report. In: *British Philosophy in Mid-Century*. London: Allen and Unwin.

Popper, K. 2005. *The Logic of Scientific Discovery.* s.l.: Taylor & Francis.(Logik der Forschung first published in German in 1935 by Verlag von Julius Springer, Vienna Austria; first English edition published by Hutchinson & Co. in 1959)

Rosenman, M. A. & Gero, J. S. 1998. Purpose and Function in Design: From the Socio-Cultural to the Technophysical. *Design Studies*, April, 19 (2).

Rowe, C. 1982/83. Program vs. Paradigm. *The Cornell Journal of Architecture,* Issue 3, pp. 9-19.

Ryd, N. 2004. The Design Brief as carrier of Client Information During the Construction Process. *Design Studies*, May, 25 (3).

Rzevski, G., Woolman, D. & Trafford, D. B. 1980. Validation of a Design methodology. *Design Studies*, 1 (6).

Sanoff, H. 1977. *Methods of Architectural Programming.* Stroudsburg Pennsylvania: Dowden, Hutchinson & Ross Inc.

Schön, D. 1984. Editorial: Design: A Process of Enquiry, *Experimentation and Research*. July, 5 (3).

Schön, D. 1984. Problems Frames and Perspectives on Designing. *Design Studies*, July, 5 (3), pp. 132-136.

Schön, D. 1991. *The Reflective Practitioner*. s.l.: Avebury The Avademic Publishing Group. (First published by Basic Books in 1983)

Steadman, P. 1979. The History and Science of the Artificial. *Design Studies*, 1 (1).

Steele, J. 2001. *Architecture and Computers: Action and Reaction in the Digital Design Revolution*. s.l.: Laurence King.

Summerson, J. 1957. The Case for a Theory of Modern Architecture. *R.I.B.A. Journal*.

Terzidis, K. 2006. *Algorithmic Architecture*. s.l.: Elsevier Ltd..

Webster's Third New International Dictionary: Unabridged. 1993. Springfield Mass. Könemann.

Windelband, W. 1901. *A History of Philosophy*. New York: Macmillan.