Reflections on design style readings.

I was amazed at the many diverse ways that the different papers approached the explanation of the design process. No two papers approached it in the same way, though there were some similarities in descriptions. It is very interesting that something that is so hard to decompose and define is the key ingredient for the architectural industry. Can you name any other industry whose functional core is so ill defined, documented and understood?

I propose that design is learned through the evaluation and understanding gained from our cumulative lifetime experiences. Depending upon what environments and stimuli we have been exposed to greatly affect the ideas that we can conceptualize. Questions: How would a blind man describe the room he was in? How would he describe a room that he has entered for the first time? How would a deaf person describe an orchestral concert vs. a rock concert?

I can only describe a design process by comparison to things that I have knowledge and experience about. Much of this knowledge / experience is from self discovery rather than formal structured education. Only by framing my experiences in terms used and documented by others can I gain a common understanding. Since we are all from such diverse backgrounds, how can we understand that which cannot be conveyed in writing?

When talking about defining style, I must break down the items presented into smaller pieces that I can compare to other things in my experience. I interpret things through my own experiences and not someone else’s.

When designing from a software engineering perspective, I try to discover / document all of the requirements about a project. They are written in a manner that leaves room for interpretation by others but distills to the essence of the need. The requirements are also rated as must have, nice to have, and if it’s not impossible.

Another common approach for software design is through the prototyping method. This is used most often when a certain esthetic or functional ease is desired. This process generates several examples of what a final product might look like without putting all of the details that make it function in place. This method allows feedback to be gathered quickly and early in the development process.

A third approach to software design is the iterative or waterfall approach. This process combines both requirements and prototyping into a structured method. Some requirements gathering occurs, some prototyping occurs and then both are evaluated internally and externally by the stakeholders for the project. Areas of improvement as well as success are documented and feed back into the process for the next iteration. By the short and fast iterations, more understanding of the problem and its possible solutions
are examined by a group of people that are sharing experiences, knowledge and growing a common understanding.

This iterative or waterfall approach most closely resembles what occurs in architectural design. People who work together for any length build a common understanding through shared experiences and understanding. Some people will put a marketing spin label on it to define, explain or justify their decisions to the outside world. If you asked each person to define the essence of the style, they would all answer with the pre-canned marketing speak or separately from their own interpretation framed through their experience.

Architectural design thinking involves the ability to consider everything one knows, discard the irrelevant; juggle, manipulate and predict those things that are relevant and to sum it all up in a manner that anyone can understand. Is this getting close to a god like description? or should it be labeled more as supernatural?

I have included for my review those passages from the readings that were significant to me. I will try to explain why I consider them important on occasion.
**Research in Design Thinking:**

Designers explore problem and solution together through drawing and modeling 
multifaceted approach

Obscurity about the requirements, practicability of provisions and misfits between them 
requirements restrict the creative process but ultimately determine the success or 
failure of the project

Solution – a requirement/provision match that contains an acceptably small amount of 
residual misfit and obscurity

Problem – found by examining it through proposed solutions 
sounds a lot like prototyping

Proposals give a very biased view 
only give the information you are sure about, don’t mention the weaknesses

Designers impose additional constraints that narrow the solution space and help to 
generate concepts 
make some assumptions to get the process rolling

Successful architects use solution conjectures early in the design process and generate or 
impose particular strong constraints 
someone must determine an initial direction for everyone to follow

Designers resist radical reformulations of solution concepts 
it’s never fun to go back and change the requirements once the software is written

Designers generate solutions with the aid of self imposed constraints 
I can only perceive things I am experienced with

Habitually treat problems as though they are ill defined 
more fun not to get concrete requirements at the beginning

Use a solution-focused cognitive strategy for problem resolution 
what can be done to solve a specific problem with least interference

Use a particular form of reasoning – different that inductive and deductive more like 
abductive 
I need to look up the differences between these

And recursive reasoning or reductive reasoning 
take data from last round, use as input to new round
The design studio:
I can understand and relate most to this article because it matches the experiences I have had.
There exists no usable science of design
  combination art, science and history
Artistry is liable to be confused with applied science; architecture is liable to no such confusion
The studio tradition builds examples of practice and critical reflection on practice into the core experience of learning architectural design
Program – a set of design requirements – a graphic description of the site
Present preliminary sketches and describe problems
Describe and appreciate the consequences, implications and changes stance toward the situation
Design domain – names of elements, features, relations and actions and norms used to evaluate problems, consequences and implications
Program / Use
Siting
Building Elements
Organization of Space
Form
Structure / Technology
Scale
Cost
Building Character
Precedent
Representation
Explanation
Norms for access, circulation and use
The designer’s moves yield systems of implications
  architectural design is like playing chess
The designer must consider not only the present choice but the tree of further choices to which it leads
  compared to a decision tree with pruning when bad choices are made
The designer evaluates his moves in a three fold way: desirability of their consequences, conformity or in violation of implications, and appreciation of new problems or potentials
The projected experience of passage through the space in order to take note of the larger relationships on which the qualities of the whole idea will depend
Engage in a conversation with the situation they are shaping
Constructs variations on themes with which he is familiar
  Is there any truly new thinking or concepts?
Make use of his past experience without reducing the new situation to features that conform to a set of familiar rules
  Don’t get caught in the same design rut
To manipulate his virtual world
  save time and money
To experiment at minimum risk
How designers think:
I felt like I had been teased after reading this article. There was no real meat for me to digest, just references to things to come.
Design situations vary not just because the problems are dissimilar, but because designers habitually adopt different approaches
Vocational design courses that begin to resemble degrees in behavioral and social sciences
Design in Mind:
The possibility of understanding the design process through scientific investigation
Has this actually been done? if so, I would like a description of how.
The scientific process is made explicit and replicable, while designers rarely reveal or
discuss their methods
Don’t give away trade secrets, that don’t have to be verified in journals
Scientists do a job that is essentially one of description; they try to tell us how things are.
Designers on the other hand, are prescriptive since their job is to tell us how things ought
to be
 Visionary storytelling at its root
Designers must not only express appropriate ideas and values but must also be usable and
work
 Concepts are nice, practicality is required
Design refers to an enormous range of activities from the highly constrained, numerical
and well defined problems
Problems defy complete description and lack the clarity of formulation
 When is description complete?
The information you need to solve them depends upon your ideas for solving them
 experience breeds ideas
Knowledge rich design requires us to have considerable amounts of knowledge beyond
that which is stated in the problem description
Design is a process in which there will be no one recognizably correct or optimal answer
 That can be prove without doubt, it is a matter of taste
We analyze the task and propose logical structures and processes that we imagine must or
should take place
 start by adding program assumptions
Most designers are at their best when designing rather than explaining
 if not, it wouldn’t be so difficult to document style
Assimilation, analysis, synthesis evaluation and communication
 sounds like an iteration in the software development cycle
Need for feedback loops or recursion
Waterfall iteration or circular recursion
False: design problems can be clearly stated
False: there are solutions which can be considered in some way optimal
Business of designing involves finding problems, understanding and clarifying objectives
and attempting to balance criteria for success
Designers often come to understand their problems through their attempts to solve them –
analysis through synthesis
Primary generator: a basic idea about the form the solution could take and a crude design
is predicates on the basis and tested
Designers bring their own concerns into the process
 Architects must consider the feeling of their users
Design usually involves making something that must work in some way well as
expressing some values or ideas
The reasoning of designers:
Type 1 failure: plan does not accomplish what was intended
Type 2 failure: when the execution of the plan causes side effects that were unforeseen and unintended and prove to be undesirable
Learning what the problems is “IS” the problem
Model of design as argumentation
Sachzwang – design or decision by the majority – releases the designer from responsibility
Changeables and invariants – rigid rules of variances (UBC)
Constraints – decided selected or self imposed and not implied, derived or logical necessities
Guarantors – unquestionable sources of reliable knowledge
Design takes place in a social context
Three tasks: to further develop theories of design, to learn more about the reasoning of designers
  Pursue empirical inquiries into how plans come about and what effects of plans are in comparison with which they intend
  Look for tools to support designers in their work