1. Important Points
   1. The purpose of design is “to produce a workable (implementable) solution to a given problem.” [209]
   2. “They key measure of the appropriateness of any [design] solution is that of *fitness for purpose*.” [209]
   3. “There is rarely (indeed, almost never) only one solution to a problem.” [209]
   4. In all design problems there are the following characteristics: “no single ‘right’ solution; many factors and constraints to be balanced in choosing a solution; no one measure of ‘quality;’ and no particular process that can ensure that we can even identify an acceptable solution!” [209]
   5. Software design is “more intractable and less well understood than other forms of design” [209] because of “the complexity of software,” [209] “the problem of conformity,” [209] “the (apparent) ease of changeability,” [209] and “the invisibility of software.” [209]
   6. “One means of acquiring the experience of others is through the use of a *software design method.*” [210]
   7. A design method has: “A representation part consisting of a set of notations,” [210] “a process part that describes how the model is to be developed,” [210] and “a set of heuristics that provide guidance on how the process part should be modified” [210] when encountering problems. [210]
   8. “Constraints … act to limit the ‘solution space’ that is available to the designer.” [210]
   9. “Designing for ‘maintenance’ … is another factor that may influence the form of the solutions that is adopted.” [210]
   10. “Simply doing the job correctly and within the resource constraints identified may not be enough to achieve fitness for purpose.” [210]
   11. “A major need for the designer is to be able to select and use a set of abstractions that describe those properties of the design model that are relevant to the design decisions that need to be made.” [211]
   12. “A design viewpoint can be regarded as being a ‘projection’ from the design model that displays certain of the properties of the design model.” [211] Four useful viewpoints are: “behavioural [sic],” [211] “functional,” [211] “structural,” [211] and “data modelling.” [211]
   13. “The three principal forms of description normally used to realise [sic] the design viewpoints are text, diagrams, and mathematical expressions.” [211]
   14. Successful diagrammatical descriptions have the following characteristics: “A small number of symbols,” [212] “a hierarchical structure,” [212] and “simplicity of symbol forms.” [212-213]
   15. “Statecharts [sic] provide a means of modelling the behavior of a system when viewed as a finite-state machine, while providing better scope for hierarchical decomposition and composition than is generally found in behavioural [sic] representation forms.” [213]
   16. Jackson structure diagrams “describe the ordered structure of an ‘object’ in terms of three classical structuring forms of sequence, selection, and iteration.” [213]
   17. The structure chart “captures … the invocation hierarchy that exists between subprogram units.” [214]
   18. “The Entity-Relationship Diagram (ERD) is commonly used for modelling the details of the interrelationships that occur between data elements in a system.” [214]
   19. The roles of “software design methods include: establishing common goals and styles for a team of developers,” [216] “generating ‘consistent’ documentation,” [216] and “helping to make some of the features of a problem more explicit.” [216]
   20. “Four widely-used [design] strategies are: top down,” [216] “compositional,” [217] “organizational,” [217] and “template.” [217]
   21. “Read-ahead, backtracking, and program inversion” [217] are “‘adaptations’ to the basic [JSP] process.” [217]
   22. “[Structured systems analysis and structured design] consists of an ‘analysis’ component and a ‘design’ component.” [218]
   23. “JSD encourages the designer to create a design model around the notion of modelling the behavior of active ‘entities.’” [218]
   24. “A problem with object-oriented methods is that they do encourage the designer to make decisions about ‘structure’ at a much earlier stage than ‘process’-oriented [sic] methods.” [219]
2. Disagreements
   1. “A problem with object-oriented methods is that they do encourage the designer to make decisions about ‘structure’ at a much earlier stage than ‘process’-oriented [sic] methods.” [219]

I disagree with the author’s assertion that making decisions about structure early in the design process is a problem. Assuming a thorough architectural design stage, the software design stage is the correct time to make decisions about structure.

1. Questions and Clarifications
   1. What are “lexical tokens?” [211]
   2. What are “first-generation design methods” [217] and “second-generation methods?” [217]