Chapter 1 – The Engineering Design Process

1. **In your own words, describe the difference between prescriptive and descriptive design processes. Cite examples of each. [R]**

   Prescriptive design processes "prescribe" an exact sequence of steps and decisions for realizing a design. There are often decisions that must be made in prescriptive processes for determining whether to move from one stage to the next, or to move to the next phase. Descriptive processes describe the general steps needed to achieve a design, but do not explicitly layout the steps which should be followed to achieve the design.

2. **Describe the relationship between the Problem Identification, Research, and Requirements Specification phases of the design process. [R/A]**

   Problem Identification, Research, and Requirements Specification are three early phases of the design process. The overall objective of these phases is to identify a problem, analyze it, and develop requirements for its solution. The Problem ID phase is where the end-user needs are determined, while further analysis occurs in the research phase. Both problem and research phases are used to develop a Requirements Specification that provides the requirements for those elements that must be satisfied in order for a successful design.

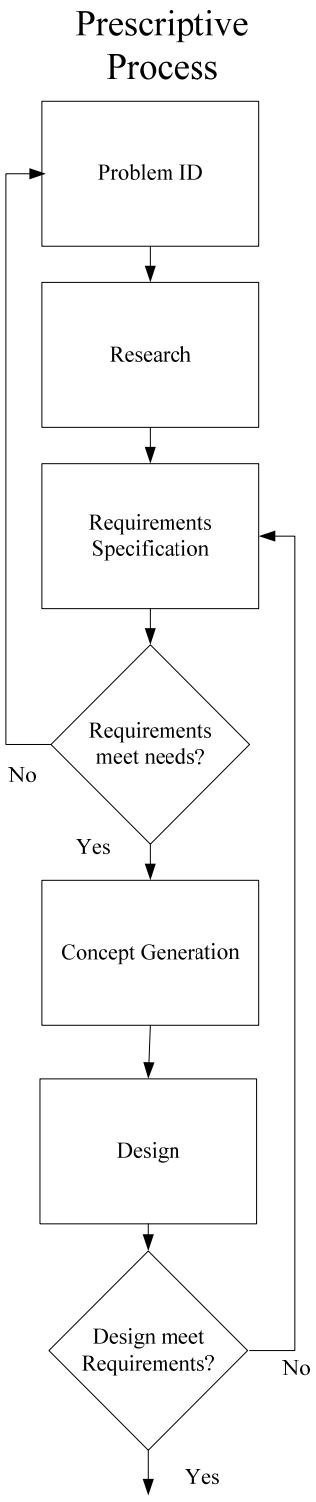
3. **Describe the relationship between the Concept Generation and Design phases of the design process. [R/A]**

   In Concept Generation, different technical options for solving the design are given – one is selected to pursue. In the Design Phase, the option selected from Concept Generation is further clarified and the design architecture is more clearly defined.

4. **Construct a prescriptive design process for the Problem Identification, Research, Specification, Concept, and Design phases of the design process. The result should be a flow chart that contains decision blocks and iteration as necessary. [A]**

   In the prescriptive design process, shown in the figure below, there are two decision points, one of which occurs after the requirements are determined. The objective in this decision is to determine whether the requirements satisfy the end-user needs. If not, the needs must be re-examined and the requirements must be updated as necessary, in order to meet the customer needs. The other decision occurs after the design is generated. Here, the objective is to determine if the design satisfies the requirements. If not, a new design concept must be generated.

   Note: *This is only one possible solution, and there are certainly other possible design processes that could be developed.*


Prescriptive Process
Problem ID
Research
Requirements Specification
Requirements meet needs?
No
Yes
Concept Generation
Design
Design meet Requirements?
No
Yes

5. Describe the main differences between the VLSI and embedded system design processes. [A]

VLSI and embedded systems design share similarities and contain differences. They are both similar in that they have phases for requirements specifications, system architecture design, and technical design. The difference between them lies in the technical design, where the steps depend upon the technology that is being developed. In the case of VLSI, steps are used to successively refine the design to meet develop a layout level circuit; however, embedded design requires that the technical design phase consists of software and hardware co-design.

6. Using the library or Internet, conduct research on the spiral software design process. What are the advantages and disadvantages of this relative to the waterfall model? Be sure to cite references. [A]

The spiral methodology reflects the relationship of tasks with rapid prototyping, increased parallelism, and concurrency in design and build activities.[1] The spiral process recognizes that errors will occur in all stages of the production process and proceeds on this basis.[2] It is agreed that the development processes will have to be revisited multiple times as the design furthers completion; therefore, unlike the Waterfall model, this methodology incorporates an iteration cycle, which is continued until the design is fully complete. A Spiral Development Model diagram can be found at http://www.hyperthot.com/pm_sdm.htm as well at other sites on the Internet.

Embedded in spiral design is the process of refactoring – changing software in such a way as to improve structure, but not affect the end result.[3] Overall, the spiral software design model is not as rigid, concrete, and strict as the Waterfall model; however, this method should still be planned methodically, with tasks and deliverables identified for each step within the spiral. The table below lists the advantages and disadvantages of the spiral design model in reference to the waterfall model.

---

1 Chapman, James. "Spiral Methodology." *Software Development Methodology.* 2005. 20 May 2005 < http://www.hyperthot.com/pm_sdm.htm >.

2 Culwin, Fintan. "The Production Process." *LAW – Learn Ada On the Web*. 1998. 20 May 2005 < http://www.scism.sbu.ac.uk/law/Section1/chap1/s1c1p3.html >

3 Hean, Daniel. "Design through to testing." *Content & Document Management System.* 2005. 20 May 2005. < http://www.yedit.com/web-content-management-system/400-design-through-to-testing.html >

	Advantages		Disadvantages
1	Increased time-to-market	1	Revisiting the same stages
2	Incremental & Iterative	2	Requirements are not fully identified
3	Promotes increase in documentation	3	Project goal is not initially established
4	No set structure or phase routine		
5	Non-idealistic		
6	Not as costly to revisit process steps		
7	Primitive to more intricate design		
8	Allows development to begin w/o full understanding		

7. Using the library or Internet, conduct research on the Extreme Programming design process. Outline the significant elements of the Extreme Programming paradigm. What are the pro and con arguments for this software development model? Be sure to cite references. [A]

Extreme Programming is a methodology – although it's debatable - of programming that was started back in the early 1990's by Kent Beck. Beck realized that there were four key components, or dimensions, that were crucial for the success of a software project – **communication**, **simplicity**, **feedback**, and **courage**. One can view extreme programming as a streamlined software development process that focuses on delivering software when needed.[4]

Characteristics of Extreme Programming

- Needs identification – get "stories" from user on what they want the software to do.
- Small teams of programmers.
- Many iterations. The team develops code and shows it to the client early to see if it satisfies and changes as necessary.
- Much early testing.
- Agreed upon coding standards are followed.
- ALL code is written by two people working together!
- Continuous Testing. All code is test early on – has test plan integrated with coding
- Coders work no more than 40 hour/ wee.
- Teamwork is encouraged and very important.

[4] Wells, Don. "The XP Philosophy." *XP Extreme Programming*. 1999. Extreme Programming. 8 Sep. 2004 < http://www.extremeprogramming.org/Kent.html >.

- Refactoring. Factor out duplicate code.

The current practices of an XP (Extreme Programmer) are subdivided into 12 arenas. These subdivisions are The Planning Game, Small Releases, System Metaphor, Simple Design, Continuous Testing (Unit and Acceptance), Refactoring, Pair Programming, Collective Code Ownership, Continuous Integration, 40-hour Work Week, On-site Customer, and Coding Standards.[5] Each of these practices is precisely defined; however, merely mentioning them is adequate for the task at hand. A flow of the Extreme Programming process can be accessed via http://www.extremeprogramming.org/ map/project.html.[6] and at other locations on the Internet.

The idea of XP seems very promising. There are some clear advantages to this style and method of programming. However, there are some very distinct disadvantages also. The table below lists all the advantages and disadvantages of this fairly new style of programming.

	Advantages		Disadvantages
1	Stresses customer satisfaction	1	Like a jigsaw puzzle (lack of structure)
2	Enables groupware style development	2	Departure from traditional methods
3	Saving money (reduce software costs)	3	Pair programming
4	Thorough testing	4	Source code hard to understand/maintain
5	Lightweight methodology (few rules)	5	Flaws with YAGNI philosophy
6	Delivers software when needed		("You Aren't Gonna Need It")
7	Doesn't overdesign desired system	6	Overlooks organizational problems

Overall, XP has many great ideas for improving efficiency and productivity; however, it still contains serious structural issues that could prove counterproductive.

> ***Note:*** *We find this (and previous problem) to be a valuable exercise because it demonstrates to students that design processes are evolving and important. The pitfall of many is to not follow a process and just try to proceed directly to a solution.*

8. Project Application. In preparation for the project and team selection, develop a personal inventory that includes a list of five favorite technologies or engineering subjects that you are interested in pursuing. Also, list the strengths and weaknesses that you bring to a project team. [P]

---

[5] Brewer, John. "Extreme Programming FAQ." *Extreme Programming (XP)*. 2001. 9 Sep. 2004 < http://www.extremeprogramming.org/rules/iterative.html >.

[6] Wells, Don. "Iterative Development." *XP Extreme Programming*. 1999. Extreme Programming. 9 Sep. 2004 < http://www.extremeprogramming.org/rules/iterative.html >.

Note: *We find this exercise an important step in starting students on the path of team formation and project selection and usually assign it on the first day of class. We setup an electronic bulletin board for the students and have them post this information publicly for the whole class to see. Students are then encouraged to review this and identify potential team-mates. We have also done a variation where each student is required to determine this information and then make a short oral presentation (2 minute pitch) to the class, in which they describe what types of projects they are interested in and what strengths/skills they can bring to a team.*