**Layered Comprehension – Interview plan**

1. Demographic questions: recent work experience, background, primary programming languages, time in project, role
2. We define “layers of hierarchy” in a software system: function/method, class/module, system. In many projects there may be one or two additional levels between “system” and “class” that is some form of collection of classes, such as “package”, “service”, or “component”.
	1. Would you add or remove a layer in the hierarchy?
	2. Give an example for a system and first level of breakdown within this system. How is that breakdown called?
3. We define a **software system** as a collection of packages that create an end-user product or end-user experience. A company may be creating a system that contains packages that are developed within the company, or imported from other companies. Alternatively, a company may be developing packages to be integrated into customer systems. *[Show printed out definition]*
	1. What are your thoughts on this definition?
4. How would you define “understanding” of a
	1. Function
	2. Class
	3. System
5. Understanding and using:
	1. Do you think it is possible to use a function without understanding it? What level of understanding is required to use a function?
	2. Same for class
	3. Same for package/service
6. Understanding and debug/maintenance:
	1. Do you think it is possible to debug/maintain a function without understanding it? What type of understanding is required to debug/maintain a function? Use examples
	2. Same for class
	3. Same for package/service
7. Mutual dependencies:
	1. Does one need class level understanding to understand a function? In a -3 to 3 scale, how much class level understanding is required to understand a function?
	2. Does one need package level understanding to understand a class? Same scale
	3. Does one need system level understanding to understand a package? Same scale
8. Understanding process:
	1. When you are given a function, what is the process for you to understand it?
	2. When you are given a class, what is the process for you to understand it?
	3. Assume you are evaluating software packages developed by another team/company for use in your system. You do not plan to modify the package, just use it.
		1. What is the first thing you would look at?
		2. How would you continue with the evaluation?
	4. Assume you are joining a team that develops a software package, or that a new developer joins the team and needs to understand the package
		1. What is the first thing you would look at?
		2. How would you continue the comprehension process?
9. Comprehension tests:
	1. What would constitute a test that would demonstrate that you indeed understand a function?
		1. Existing examples may include: using the function; adding a feature; debugging; rewriting code from memory; describing function flow; describing the function interface; describing how it is used; fill in the blanks. Which of those are most applicable for testing function comprehension?
	2. Same for class
	3. Same for package
	4. Same for system
10. Organize the 6 cards to reflect the importance of each element for the comprehension of: *[cards are: API documentation; design document; sample code demonstrating use; source code; inline documentation and function header documentation; training material]*
	1. Function
	2. Class
	3. Package for evaluation
	4. Package for maintenance
11. Personnel:
	1. How many developers are typically responsible for the development of a package?
		1. How many of those developers have a package level understanding?
	2. How many developers develop the system?
	3. Who are the people in the organization that have package understanding? What roles do those people have?
	4. Who are the people in the organization that have system understanding? What roles do those people have?
12. Give examples:
	1. Examples of a function where there was a problem understanding it and how it was resolved
	2. Examples of a class where there was a problem understanding it and how it was resolved
	3. Examples of a package being evaluated and the evaluation process
	4. Examples of a package-level issue (cross classes) and how it was addressed

**Architecture**

The following questions refer to “architecture” of a specific project (as opposed to architecture paradigms)

1. What is a “system architecture”?
2. What is your level agreement with the following sayings? Rank between -3 and 3, where 0 means neutral, +-2 mean “agree/disagree”, +-1 mean “leaning to agree/disagree”, +-3 mean “strongly agree/disagree”. You can differentiate between existing state and desirable *[follow-up questions for interesting replies]*
	1. Architecture is the structure of the system
	2. Architecture reflects design decisions
	3. Architecture defines only the internal structure of the system
	4. Every system has an architecture
	5. Implementation changes over time as a result of architecture changes
	6. Architecture defines only the externally visible properties of the system
	7. Architecture is defined in documents
	8. Architecture is a UML description of the system
	9. Architecture defines internal interfaces between components in the system
	10. Architecture changes over time as a result of implementation changes
	11. Architecture reflects the company’s organizational structure
	12. Architecture defines internal interfaces between teams in the development group
	13. Everyone in the development groups is familiar with the architecture
	14. Architecture is the product of a design process
3. Would you like to restate your definition for “system architecture”?
4. How is system architecture developed?
	1. Dedicated person that is responsible for the architecture?
	2. Group effort?
	3. Architecture documents? What do they include? UML? Diagrams?
	4. Other ways of communicating the architecture? Forums? Emails?
5. Is it important to understand the architecture? Why?
6. Is system understanding the same as architecture understanding?
7. Any additional related comments?