**cs4473B/cs9551B**

**INDIVIDUAL TEMPLATE**

**Reading Summary and Questions and Answers**

**Rules – please note these carefully:**

* Submission filename MUST be: “**Last-name First-name” “Group”<id>\_”Chapter” (or reading) <id> (e.g., Blogs Joe\_Group 3\_Chapter 2)**
* This template must be used for ONLY ONE chapter (or reading) at a time. For the second item of reading, if any, please use another copy of this template.
* **PLEASE compress multiple files (one file/chapter) using standard (e.g., Windows) compression that can be uncompressed on a Windows machine with simple clicks. Please do not use unusual/fancy compression tool; your template won’t be graded and you will be penalised.**
* **Submission to be done on OWL as per the deadline set.**
* **EMAIL SUBMISSION WILL NOT BE ACCEPTED AND WILL INCUR PENALTY.**
* **Use of template is mandatory**: submission of text made outside the template will result in a zero mark.
* **Altering this template (meta-items) will incur a penalty.**
* Submission format is **MS WORD only** **(not PDF)**. PDF files will incur a penalty.
* The source of the answer captured from the chapter must be accurate or closest to the context (e.g., Chapter #, Section or sub-section #, page number, etc.).
* The question must be properly and fully specified, and easily understandable. Cryptic text or grammatical errors will be penalised – no appeals accepted.
* The question must not be so general or non-specific to apply to non-specific answers.
* The answer (text identified from chapter) must be an important point, not something trivial or highly specific to a context.
* Answer from the book must be copied “as-is” from the text (**reference** to chart/table/figure/etc., in the reading is permitted and encouraged). Cryptic text or grammatical errors will be penalised – no appeals accepted.
  + *If in doubt about the quality or acceptability of your text, you will have one chance to have it reviewed by the instructor for “live” feedback. No emails please.*
* Assessment of your submitted template will be done sometime before the end of the term. Please do not expect marks of your submission every week!
* There will be no “remake” of the summaries (e.g., for improving the mark). It is a one-time submission and assessment. Late submissions will not be accepted.

**Part 1: Summary**

|  |
| --- |
| **Group member’s name**: Yulun Feng **Group No**: 2 |
|  |
| Please write the **full reference** of the reading in the WHITE box below.   * + Chapter #, Chapter title (or article title if appropriate).   + Book title   + Author(s)   + Publisher   + Book edition, Year of publication   (Example shown below; overwrite on that space.) |
| Chapter 14: Requirements development  *Software Requirements, 3rd ed.*  Weigers and Beatty  Microsoft, 2013 |
| Please write in the WHITE box below an abstract of the reading in **50-75 words**. |
| **Chapter 14 covers non-functional requirements in software developments, showing quality attributes such as usability, reliability, performance, and security. These non-functional requirements are important roles in project. Also, the chapter gives approaches to identifying, prioritizing, and specifying these attributes. And it also discussed the point of it is essential keeping the balance between specific project need and user satisfaction need.** |

**Part 2: Questions, Answers and Comments**

|  |
| --- |
| Please create **THREE** important **Question-Answer-Comment sets** from the given reading:   * State your question succinctly. Add more lines as needed. * State <LOCATION of answer in the book (e.g., section #>: <as-is answer EXACTLY from the book> ; please indicate whether a hardcopy or ebook ‘cause the page numbers are not the same. * State your Comment related to the answer; this is mandatory.   **NOTE: The following will be penalised: a cryptic question; inaccurate copying of the answer; a trivial point; “too specific or small detail”; missing or trivial comment; etc.. Informative, insightful, or key concept/idea, question, answer, and comment are expected.**   * + Actual chart/table/figure/etc. must NOT be given in the text below, but you may site its location in the reading by giving precise specification. |
|  |
| For staff use only:  **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**  **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_** |
| **(1)**  **QUESTION: How should we balance conflicting quality attributes in software?**  **ANSWER:**  **<ebook><p264> Step 3: Prioritize the attributes - “Prioritizing the pertinent attributes sets the focus for future elicitation discussions.”**  **<ebook><p265> Figure 14-1**  **<ebook><p265> Trap – “Don’t neglect stakeholders such as maintenance programmers and technical support staff when exploring quality attributes. Their quality priorates could be very different from those of other users.”**  **COMMENT (also include where possible: an \*example\*, citation, justification, etc. -- to support your comment).**  **Balancing conflicting quality attributes in software demands a deliberate approach that prioritizes the most important attributes, as this sets the tone for future conversations and choices. It is critical to consider the perspectives of diverse stakeholders, such as maintenance programmers and technical support workers, whose needs may differ greatly from other users. Integrating their views guarantees a comprehensive grasp of quality requirements, resulting in a software solution that is strong, efficient, and meets the various expectations of all users. Furthermore, the reference to Figure 14-1 in the chapter most likely gives a visual framework to aid in this complex prioritization process."** |
|  |
| For staff use only:  **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**  **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_** |
| **(2)**  **QUESTION: How can robustness in software systems be achieved while ensuring user-friendliness and error recovery?**  **ANSWER:**  **<ebook><page 275> Robustness – “Usability encompasses several subdomains beyond the obvious ease of use, including ease of learning; memorability; error avoidance, handling, and recovery; efficiency of interactions; accessibility; and ergonomics. ”**  **<ebook><page 279> Usability – “Usability encompasses several subdomains beyond the obvious ease of use, including ease of learning; memorability; error avoidance, handling, and recovery; efficiency of interactions; accessibility; and ergonomics. ”**  **COMMENT (also include where possible: an \*example\*, citation, justification, etc. -- to support your comment):**  **The junction of resilience and user-friendliness in software is dependent on a complete usability methodology. Key factors such as mistake prevention, handling, and recovery, as explained on pages 275 and 279, are critical. These aspects not only improve the software's error-handling capabilities, but also the overall user experience. A strong system must anticipate and elegantly handle user mistakes, ensuring smooth interactions and accessibility. Integrating these elements results in software that is both durable and intuitive, catering to a varied user base. The challenge is to strike a balance between technical robustness and user-centric design, assuring consistent performance and engaging interaction inside a small, efficient framework.** |
|  |
| For staff use only:  **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**  **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_** |
| **3)**  **QUESTION: What are the challenges and strategies in ensuring scalability in software development, and how does it impact other quality attributes like performance and modifiability?**  **ANSWER:**  **<ebook><p263> Table 14-1**  **<ebook><p281> Efficiency – “Efficiency is closely related to the external quality attribute of performance. Efficiency is a measure of how well the system utilizes processor capacity, disk space, memory, or communication bandwidth. If a system consumes too much of the available resources, users will encounter degraded performance.”**  **<ebook><p285> Scalability – “Scalability has both hardware and software implications. Scaling up a system could mean acquiring faster computers, adding memory or disk space, adding servers, mirroring databases, or increasing network capacity. ”**  **COMMENT (also include where possible: an \*example\*, citation, justification, etc. -- to support your comment):**  **Scalability in software design, as covered in Chapter 14, requires important trade-offs between performance and resource utilization. The "Efficiency" section (pages 280-281) emphasizes the importance of optimizing resource consumption, such as CPU capacity and memory, to guarantee that the system scales successfully and without performance loss. Meanwhile, the "Scalability" section (pages 284-285) discusses the issues of preparing for rising user or transaction volumes. This demands creating flexible algorithms and structures that can handle growth while remaining efficient. These findings highlight the delicate balance needed in scalable software architecture to ensure it remains durable and performs reliably as it increases.** |