Name : Monica Gloudemans Date: 11/21/2011

(Type answers and include question/only brief answers required/1-3 sentences each subtopic)

CS 560 Midterm I: Software Engineering (Fall 2011)

1. **[10 pts] We have covered the architectural hierarchical decomposition process from your initial vision document. Describe:**
    a. **Application architecture** Application architecture is the organizational design, the "blueprint", of an entire software application, including all components and sub-components, interfaces, and external applications that communicate with the system. It is a high-level view of the system.
    b. **Information architecture** Information architecture refers to the analysis and design of the data stored by information systems, concentrating on entities, their attributes, and their interrelationships. The design matrix (FR vs. DP matrix) was used to describe the information architecture for this project.
    c. **Systems architecture** The system architecture describes how the components of a system are organized and integrated. The system architecture is represented in the form of the FRs, DPs, PVs, tree diagram, the module-junction diagram, and the flow diagram.
    d. **UML architecture** The UML architecture models the system from different perspectives including a design, implementation, process and deployment perspective. It includes static views and behavioral views. Our project included class, component, activity, sequence and use-case diagrams to describe our system.

2. **[10 pts] In RUP, we are leveraging the axiomatic design process in the inception and elaboration phase. Describe the requirements engineering and elicitation process in more detail from initial gathering of VOC to CN to FR in axiomatic design, including risk concepts.**
The first step in the requirements engineering and elicitation process is to listen to the Voice of the Customer (VOC). This may involve interviews, questionnaires, focus groups, apprenticing and modeling. Next the customer's needs are identified by defining the client's perception of the problem and problem domain, the functions, constraints, and other information that may be relevant to the system to determine what software must be created to solve the customer's problem. High risk elements are addressed early in the process so risk can be designed out of the system. Finally the CNs are translated to functional requirements for the system following the best practices for specifying requirements including correctness, completeness, consistency, trace-ability, prioritization, unambiguousness, testability, and modify-ability.

    a. **Design matrix results in which UML diagram type?** Class and Object diagrams.
    b. **DSM results in which UML diagram type?** Component diagrams.
    c. **What is the V-Model?** The V-Model is the axiomatic model for designing software. The first step is to follow a top-down approach (the left leg of the V) and build the software hierarchy (involves identifying customer needs, define FRs, mapping and decomposition). Next the full design matrix is generated to define modules or

components (bottom of the V). Finally the object-oriented model is built from the bottom up (the right leg of the V) following the axiomatic design flow chart (identify classes, establish interfaces, coding with system architecture).

d. **QFD** The 3 main goals in implementing Quality Functional Deployment (QFD) are: prioritize spoken and unspoken customer wants and needs, translate these needs into technical characteristics and specifications, build and deliver a quality product or service by focusing everybody toward customer satisfaction. In ACS 560, we used a house of quality to perform the first phase of QFD analysis for our project.

e. **FMEA** A failure modes and effects analysis (FMEA) is a procedure in product development for the analysis of potential failure modes within a system. The potential failures are classified by the severity and likelihood of the failures based on past experience with similar products or processes. The goal is to design the failures out of the system. Failure modes are any errors or defects in a process, design, or item, especially those that affect the customer, and can be potential or actual. Effects analysis refers to studying the consequences of those failures.

3. **[5 pts] We have covered how to establish class architecture first as a foundation to detailed design with UML. When using Axiomatic Design process to develop classes, how do the following map to OO Design object elements [hint: see paper handout on "Object Oriented Design with Axiomatic Design" :**

   a. **FR** FRs map to classes or objects.
   b. **DP** DPs map to attributes. The DPs associated with an FR are the attributes of the object defined by that FR.
   c. FR/DP design matrix intersection The relationships between the FRs and DPs define the methods of the object defined by the FR.

4. **[5 pts] Describe the purpose of these tools/software used in your project:**
   a. **Acclaro DFSS** The Acclaro DFSS tool was used to list the functional requirements (FRs) and design parameters (DPs). The FRs and DPs were then decomposed with the goal that each FR would represent one piece of information and that each DP would be associated with one FR. The design matrix was analyzed to visualize dependencies in the hopes of being able to reduce them through removing or reassigning some DPs. The design matrix was used to represent the information architecture and the DSM was used to represent the system architecture of the project. The Acclaro too also provided a tool for constructing a House of Quality analysis and a FMEA analysis.
   b. **Visio** Visio was used for modeling and creating diagrams. The application architecture drawing as well as all of the UML diagrams were constructed using Visio. UML templates were very helpful in designing the diagrams.
   c. Basecamp Basecamp is a tool for collaboration on projects. It facilitates file sharing, assigning responsibilities, tracking milestones and events, communication between members of the team and overall project planning. My partner and I did not use this

tool; however, if the group had been larger than 2 members, Basecamp would have been a very useful tool.

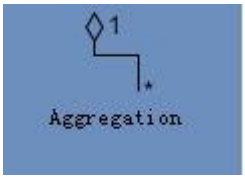
d. **MS Projects** Microsoft Project was used to construct and maintain the Gantt Chart to depict the project's past progress, plan future work and assign responsibilities.
e. **MS PowerPoint** Microsoft PowerPoint was used to create presentations documenting artifacts and progress during the inception phase, at the beginning of the elaboration phase, and at the conclusion of the course.

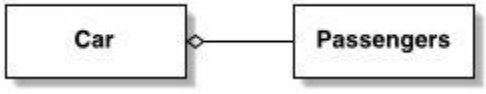
5. [5 pts] Describe the standards applied in this project

a. **IEEE-830 SRS** This standard describes a recommended practice for writing a software requirement specification (SRS). It describes the content and qualities of a good SRS including functionality (What is the software supposed to do?), external interfaces (How does the software interact with people, the system's hardware, other hardware, and other software?), performance (What is the speed, availability, response time, recovery time of various software functions, etc.?), attributes (What are the portability, correctness, maintainability, security, etc. considerations?), and design constraints imposed on an implementation. (Are there any required standards in effect, implementation language, policies for database integrity, resource limits, operating environment(s) etc.?)
b. **IEEE-1058 PMP** This standard specifies the format and contents of software project management plans. The elements that should appear in all software project management plans are identified. These include the project's motivation, objectives, success criteria, major deliverables, and constraints. It also defines interfaces to entities outside of the project, identifies the internal project structure, and defines roles and responsibilities for the project. Finally, managerial and technical process plans include resource acquisition, cost estimation, risk management and project close out plans.
c. **IEEE-1016 SDD** This standard specifies requirements on the information content and organization for software design descriptions (SDDs). An SDD is a representation of a software design that is to be used for recording design information, addressing various design concerns, and communicating that information to the design's stakeholders. The SDD often incorporates diagrams to model the structure and behaviors of the system.

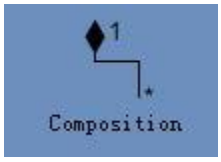
6. [5 pts] In OO design, describe the concept [with symbol]:

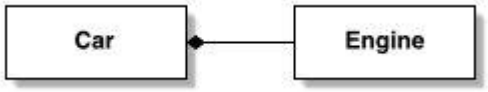
a. Aggregation Aggregation is a type of association that specifies a whole/part relationship between the aggregate (whole) and component part. This relationship between the aggregate and component is a weak "has a" relationship as the component may survive the aggregate object. The component object may be accessed through other objects without going through the aggregate object.





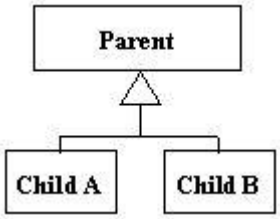
b. Composition Composition is a type of association very similar to aggregation. The relationship between the composite and the component is a strong "has a" relationship, as the composite object takes ownership of the component. This means the composite is responsible for the creation and destruction of the component parts. An object may only be part of one composite. If the composite object is destroyed, all the component parts must be destroyed.



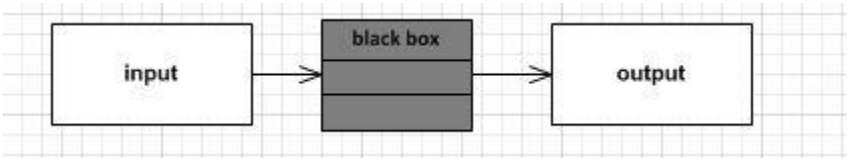


c. Polymorphism Polymorphism is when the same operation or method is implemented in different ways in different subclasses.

d. Inheritance Inheritance refers to an "is-a" relationship where the derived class (child class) inherits common functionality (methods and/or attributes) from the base class (parent class.

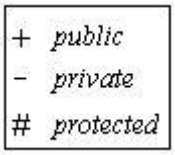


f. **Black box** A black box is a device, system or object which can be viewed solely in terms of its input, output and transfer characteristics without any knowledge of its internal workings-- its implementation is "opaque".

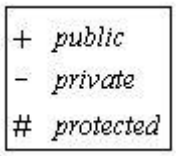


5. [5 pts] In OO design, describe the concept [with symbol]:

a. Public operations Attributes and operations which may be used by other classes.
Symbol +

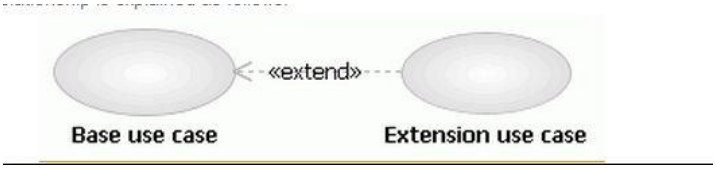


b. Private operations Attributes and operations which may only be used by class members. **Symbol -**



c. Inclusions An including use case calls or invokes the included one. Inclusion is used to show how a use case breaks into smaller steps. The included use case is at the arrowhead end.

d. **Extensions** An extending use case adds goals and steps to the extended use case. The extensions operate only under certain conditions. The extended use case is at the arrowhead end.



6. [5 pts] You were given a handout on software project management by Fairley. Describe how the following concepts are used in your project.

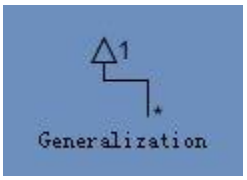
a. Architecture Decomposition View (ADV) The Architecture Decomposition View is concerned with specifying the software modules, their interrelationships and assigning specific functional requirements to the appropriate modules. The ADV for our project consists of the four architectures described in Question 1.

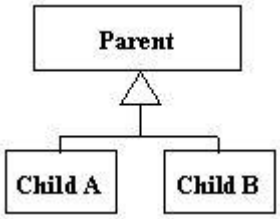
b. Work Breakdown Structure (WBS) A work breakdown structure is the assignment of activities and tasks into units with clearly defined roles, responsibilities, and managing authorities. Implementation is not within the scope of the course requirements and as such, we have not constructed a work breakdown structure. However, the decomposition view of the software was structured so that concurrent work assignments could be made to future developers. For example, four basic application components were designed to provide the main functionality of the system. Because these modules were designed to be independent of each other interacting only through interfaces, development of the modules can between distributed amongst the software developers.

7. **[5 pts] In Object oriented (OO) design, describe the concept [with symbol]:**

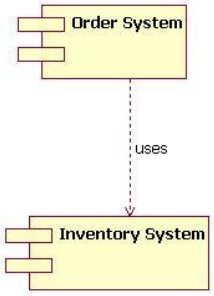
a. Association An association is a relationship linking two or more classes or objects.

b. Generalization A class (child class) that inherits attributes and operations from a more general class (the parent class).

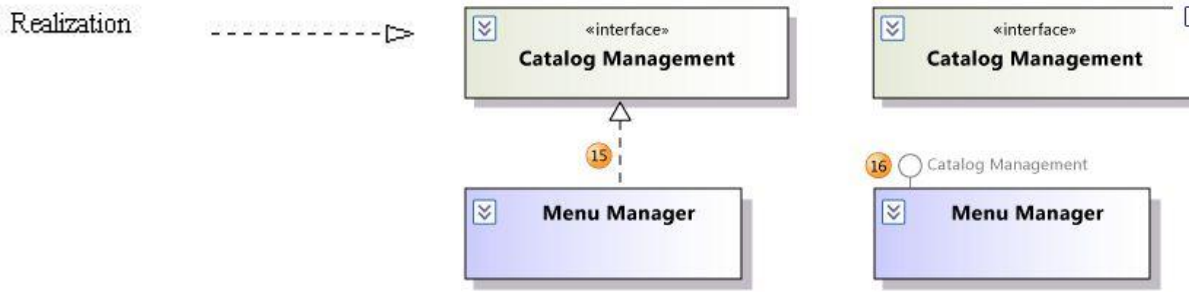




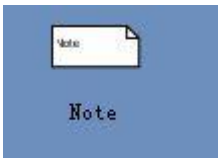
c. Dependency One class uses another.



d. Realization The relationship between a class and its interface. The class guarantees to implement the operations of the interface.

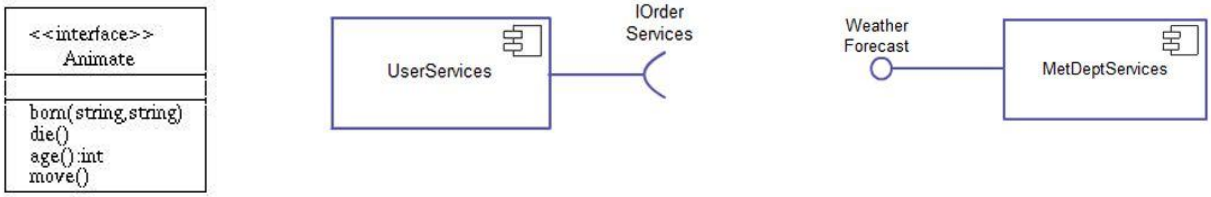


e. Annotation A note that enables you to attach comments, constraints, requirements, and graphics to a UML diagram.



f. Interface A collection of operations that a class carries out.

required interface provided interface



8. [5 pts] Define each performance attribute:

a. Efficiency can be measured indirectly by measuring the amount of time (execution efficiency) or storage (storage efficiency) needed when running the software through a particular compiler, under a specific OS, on a designated hardware architecture.

b. Flexibility the ease with which the software can be used in various situations and environments including expandability, generality, self-descriptiveness, modularity

c. Integrity is the ability of a system to withstand attacks to its security.

d. Security is the degree of protection against danger, damage, loss, and crime. It must take into account the actions of people attempting to cause destruction.

e. Maintainability is the ease with which software can be maintained in order to isolate defects or their cause, correct effects or their cause, meet new requirements, make future maintenance easier, or cope with a changed environment.

f. Portability a set of attributes that reflect the capability of software to be transferred from one environment to another. Includes self-descriptiveness, modularity, machine independence, software system independence

g. Reliability is the capability of software to maintain its level of performance under stated conditions for a stated period of time. It is also defined as the probability of failure-free operation.

h. Usability is an attempt to define user friendliness. It can be measured in terms of physical and intellectual skill required to learn the system, time required to become moderately efficient in the use of the system, the net increase in productivity over the system it replaces, or a subjective assessment of users' attitudes towards the system.

9. [5 pts] Describe how you have used your concept map and basecamp tool to organize your work as a team using RUP as a guide. We did not use the basecamp tool for this project. The CMAP tool was very useful for organizing the artifacts of our project. Each

team member maintained her own personal CMAP to document individual progress throughout the course. A shared team CMAP was used to divide the project into the 2 main phases—inception and elaboration. Artifacts associated with a specific phase could then be linked to the appropriate CMAP. The CMAP was essential in maintaining a historical record of each document, was simple to back up and was a convenient tool for posting the most current version of each document in an easily accessible location. The CMAP link could be accessed for class presentations as well.

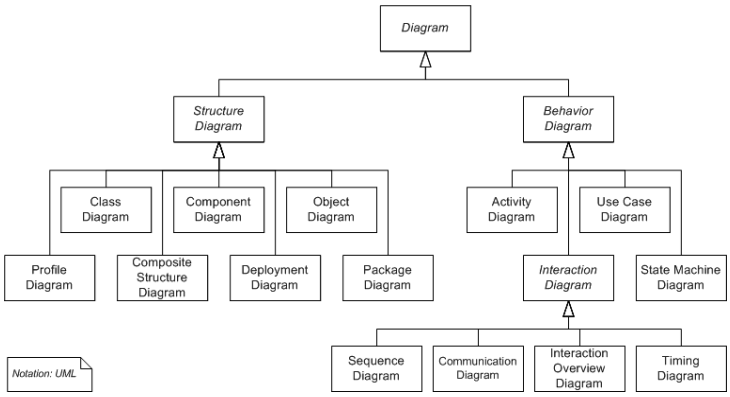
10. [5 pts] You are working on SWEBOK reviews KA-1 through KA-11. Which three areas apply to your role in the project and why?

a. Area 1 ***SWEBOK KA-1 Software Requirements*** This KA addressed the elicitation, analysis, specification, and validation of software requirements which was a key facet of the inception phase of our project. The KA was useful in defining the qualities of good requirements which we tried to follow: requirements should be stated clearly, quantitatively, and unambiguously. The KA also described another aspect of our project—architectural design and requirements allocation; specifically the process of identifying the components that would satisfy the requirements and describing the architecture from multiple viewpoints.

b. Area 2 ***SWEBOK KA-2 Software Design*** This KA addressed the steps and concerns in designing software which has been the emphasis during the elaboration phase of our project. Particularly relevant were the sections concerning architectural design—describing the software's top-level structure and organization and identifying the various components (axiomatic design phase) and software detailed design—describing each component sufficiently to allow for its construction (currently in progress with the SDD)

c. Area 3 ***SWEBOK KA-9 Software Engineering Tools and Methods*** This KA describes the types of computer-based tools that can assist in the software life cycle process and also software engineering methods that can add structure to the process. By using software development tools, the repetitive actions can be automated and the administrative load reduced allowing the software engineer to concentrate on being more creative. We used modeling tools, project management tools, risk management tools, and heuristic methods as described in this KA.

11. [10 pts] Other than class and component diagrams that all teams are required to use, select 3 UML behavior diagram types that your project could use and why?



Our team chose to use the following additional behavioral UML diagrams:

Use case diagram—to model all of the types of interactions between our system all of the potential users of the system. This gave us a high-level overview of the behaviors that our system would need to implement to satisfy the needs of all user types.
Activity diagram—to model the flow control and high-level interactions between the user(s) and the system. This gave us a means for thinking about what types of operations and data structures our system would need to implement.
Sequence diagram—to model the detailed operation, data structures, and interactions between components that are used for a single use-functionality (for example, "take an assessment"). This gave us a means for analyzing our class diagrams (attributes, interactions and interfaces) to ensure that they were complete and addressed all of the functionality required for the system.

12.[5 pts] We reviewed UML and related concepts for automation in software systems engineering.

a. What is the relationship between UML and SysML? SysML is visual modeling language used for modeling systems that include both hardware and software components. It is defined as a UML Profile, which allows it to reuse the notation and semantics of UML 2.x . SysML includes 7 of the UML diagram types (Activity, Class, Composite-Structure, Package, Sequence, State, and Use- Case diagrams) as well as 2 additional unique diagrams (the Requirement diagram and the Parametric diagram).

b.What is executable UML? Executable UML or xUML, is a modeling language that graphically specifies a system at a level of abstraction that is independent of programming language and overall software organization. The models are testable and can eventually be compiled from platform-independent models into platform-specific models that generate executable code. The advantages of using xUML include increased reuse of software, lower time and cost of software development and cross-platform functionality.

13. [5 pts] Describe how this course has helped you organize your team

a. Management This course introduced us to the tools of Microsoft Project, the CMAP and project management in general. Constructing the Gantt chart helped our team plan the phases of the project, verify that we were completing tasks on schedule, and fairly distribute responsibilities. The course required frequent, clear communication between team members so that all work was completed and not duplicated needlessly.

b. Architecture The course required four levels of architecture to be developed, each providing a unique view of the project and requiring different types of analysis to construct. The axiomatic design process ensured that each functional requirement was satisfied by at least one system component and provided an easy way to construct class and component diagrams. The UML/class architecture modeling design aided in a more complete understanding of how the software could actually be implemented.

c. Detail design Both the axiomatic design process and the UML modeling steps were instrumental in guiding our thoughts about the details of the project design. We were able to design at a level that included details of the functionality of our product separate from specific technologies and specific implementations.

d. Documentation The Vision Document was essential to mapping out the initial project idea. It enabled us to identify the stakeholders, customer requirements and product features that then drove the writing of the 3 major documents required for the course: SRS, PMP and SDD. These 3 required documents gave us a feel for several of the major components in any software engineering process. All of them were based on IEEE standards (830, 1058, 1016 respectively) which added a level of validity to the software engineering process. The CMAP was a very useful tool for maintaining the history and the current documentation of the project artifacts in an easily accessible location.

14.[5 pts] Cyberphysical systems: Describe

a. What is it? A cyberphysical system is a system designed as a network of tightly interacting physical and computational elements.

b. Why is it important? Advances in the links between computational and physical elements will dramatically increase the adaptability, autonomy, efficiency, functionality, reliability, safety, and usability of cyber-physical systems. These systems will increasingly deliver more sophisticated capabilities in a broader range of areas such as precision (robotic surgery), dangerous operations (search and rescue operations), coordination (traffic control), augmentation of human capabilities (increased visual acuity). Most of the major developing countries are investing large amounts of money in the area of cyber physical systems; the United States must also or it will be left behind.

15. [5 pts] Software Engineering: Identify a topic of interest in

a. Past I am interested in object-oriented system and software design. Particularly the modularity aspect of it, and how it has impacted the software engineering process.

b. Present I am interested in learning more about the engineering of "systems of systems". Specifically how distributed systems, middleware, and open-source software are impacting the software engineering process with regards to meshing interdisciplinary components (for example, medical devices, intelligent agents, mathematical modeling, computer-human interfaces, privacy issues, security issues).

c. Future I am interested in software engineering applied to biomedical problems at the molecular level. I am intrigued by the studies such as the following taken from my ACM weekly news update:

Easily 'Re-Programmable Cells' Could Be Key in Creation of New Life Forms *University of Nottingham (United Kingdom) (11/07/11) Emma Thorne*

Researchers at the University of Nottingham working on the Towards a Biological Cell Operating System (AUdACiOuS) project are developing an in vivo biological cell-equivalent of a computer operating system. The researchers say the project, which aims to create a re-programmable cell, could revolutionize synthetic biology and lead to completely new and useful forms of life using a relatively hassle-free approach. "We are looking at creating a cell's equivalent to a computer operating system in such a way that a given group of cells could be seamlessly re-programmed to perform any function without needing to [modify] its hardware," says Nottingham professor Natalio Krasnogor.

Bonus: [10 pts] List up to 10 aspects of this course you enjoyed/learned from the most?

I have had several negative experiences working on group projects in the past; however, the team experience that I have had during the course of this project has been very positive.

The course required collaboration with frequent communication, the development of a strategy for division of labor, and a plan for keeping documents updated. I believe my experiences throughout this course have strengthened my skills in these areas.

I gained experience using the CMAP tool for organizing artifacts and structuring my thought process as we proceeded through the project.

My teammate and I were able to develop a rapport that allowed both positive feedback and constructive comments. I have gained not only a peer whose expertise, work ethic and problem-solving skills I value and trust, but also a new friend.