*Posted on February 24th, 2014*

**COMP 361 – Developing Diagrams for Systems Analysis**

The 5 diagrams mentioned in the study guide are: use case diagrams, system sequence diagrams, activity diagrams, state machine diagrams, and problem domain class diagrams. Each have their own unique purpose in the process of systems development.

The use case diagram is developed to visualize and summarize the ways that users and other actors interact with the system. They can be used for either the traditional or object-oriented approach. However, they do not always need to be developed. This is because the use case diagram is similar to the event table. Sometimes the event table can be used in place of the use case diagram and vice versa. However, in the user goal and CRUD techniques, the use case diagram is an essential part of the process.

A system sequence diagram (SSD) is an interaction diagram with more of an emphasis on how the user actually interacts with the system. This emphasis is achieved by showing both input and output data to the system, versus showing the different objects that users interact with. This diagram is used together with use case descriptions to detail individual use cases. It is not always essential, but it does provide a valuable means of quickly assessing how a use case works. This is more valuable for complicated or detailed use cases, and need not be done for all use cases. A fully developed use case or activity diagram is also required, so if these are not already developed, then constructing the SSD should not be done.

An activity diagram is used to describe the work-flow for a user request or other business transaction. It shows the steps taken from beginning to end throughout the process. It is typically used with the object-oriented approach, but it can be used with any of the other development approaches as well. the focus is on the actual sequence of activities, so it is direct and simple. It is a good graphical representation to use with users to check for understanding of the business processes of a system. This use of the diagram as a communication tool makes it valuable, but for simple work-flows, the process can be written down as a description. However, because the diagram is more easily understood and relatively simple to develop, the should be created for almost all situations. This makes reviewing system requirements with users much easier.

The state machine diagram is used to show the state of objects in the system. For example, a shipment may be pending, shipped, or closed. Other objects in the system may be in a number of states at any time. The state machine diagram visualizes the process of transitions between states. It is very beneficial to develop this diagram because it helps analysts think critically about the system and can lead to new questions or concerns about the system. Even for small systems, a state machine diagram is important to develop. No other type of documentation or diagram reliably captures the same information as well as the state machine diagram.

The domain model class diagram is another important diagram for systems analysts. It shows all the classes of objects for a system. However, it is mainly used with an object-oriented approach. The entity-relationship diagram (ERD) is typically used in the traditional approach, but this division is changing, with many developers now using the domain model class diagram in place of the ERD. This diagram is absolutely essential in analyzing system requirements because it shows all the objects that will need to present in the new system. While this information can be gleaned from an event table or use case descriptions, this model is a much simpler reference than having to slog through a large amount of text. Even for extremely small systems, this model is always a must.