### 1. As software becomes more pervasive, risks to the public become an increasingly significant concern. In your own words, develop a realistic scenario where the failure of a software could do great harm to humans (economically or otherwise).

In this week's readings, we read several case studies regarding the public safety concerns as software becomes more ubiquitous in our lives. Specifically, we learned about the safety issues concerning personal insulin pump software, mental health patient record management systems, and weather station software (Sommerville, 2011). I conducted some research to look for other ways in which software failure can cause great harm to humans and came across a truly scary example: nuclear plants (Carvalho, 2010).

The article I found described how a nuclear power plant in Brazil was inoperable due to software complications. Additionally, it had been offline due to an "automatic shutdown after a malfunction in the reactor's electronic protection system” (Carvalho, 2010). Imagine if the system did not have an automatic shutdown protocol, or if that software had failed.

A malfunction dealing with a nuclear reactor could have devastating consequences to humans and the environment. Perhaps you remember as I do the nuclear accident at Chernobyl in 1986, in which 31 people died. The costs of that nuclear meltdown continue today in terms of cancer risk from the radioactive exposure both to the over 500,000 workers and many more local inhabitants (Wikipedia, 2015). It is imperative that software developed for nuclear plants not fail.

**2. "The only deliverable work product for a successful project is the working program". Comment**

"Deliverables" specifically means whatever is delivered to the customer. With this definition in mind, a working program is *not* the only deliverable when creating a successful software development project. Rather, there can be many deliverables generated during the development of software.

While the deliverables can include the actual source code, it can also include other products such as user manuals and operating procedure manuals or other sources of documentation (Aggarwal & Singh, 2005).

Moreover, deliverables can consist of individual parts of programs, rather than an entire working program. Especially in this era of object oriented programming, even simple subroutines can provide value. Therefore, an independent program module, which performs a specific function, can be considered a deliverable (Aggarwal & Singh, 2005).

**3. What are the differences between Generic Software development and Custom Software Development? Which one would you advocate for and why?**

As our reading describes (Aggarwal & Singh, 2005), software products can be separated into two categories: Generic Software and Customised Software.

Customised Software, which encompasses about 80 percent of the software development projects Aggarwal & Singh, 2005), is developed with a particular customer in mind. That is, a specific customer's needs and requirements are determined. Then, a software product is designed and developed to meet those personalized customer needs.

On the other hand, Generic Software, is developed without a specific customer in mind. Rather, the target can be an entire market of consumers, or indeed the world (Aggarwal & Singh, 2005). A variety of software solutions are developed for the anonymous consumer. For example, operating systems, word processor, games, and development tools can be developed and then sold to various customers all over the world.

Of the two methods of development, Generic Software development makes the most sense in terms of generating profit per lines of code. However, Generic Software development *may not* be the best method. For some applications, the personalized touch of Customized Software development is needed, especially if the customer's specific needs are not met by the features of previously developed Generic Software solution.

**References**

Aggarwal, K.K. & Singh, Y. (2005). Software Engineering: Programs, Documentation, Operating Procedures. Ed 2. Retrieved from<http://books.google.co.in/books?id=ytdKQGJ8f_AC&printsec=frontcover#v=onepage&q&f=false>

Carvalho, J. (2010). "Software failure keeps Angra I nuclear plant offline - Electronuclear”. BNAmericas: Business insight in Latin America. Retrieved from<http://www.bnamericas.com/news/electricpower/Software_failure_keeps_Angra_I_nuclear_plant_offline_-_Eletronuclear>

Sommerville, I. (2011) "Chapter 1 - Introduction: Lecture 1 -2.” Software Engineering, Ed. 9. Retrieved from<http://my.uopeople.edu/mod/resource/view.php?id=57324>

Wikipedia (2015). "Chernobyl disaster.” Retrieved from<http://en.wikipedia.org/wiki/Chernobyl_disaster#Effects>