Q1.
Do some web research and write approximately 250 words on each of the following software technology luminaries. For each include something about their background and why they are considered leaders in the field. List some of their most important publications.

Please do not simply cut and paste sentences from the web. I’m looking for your voice.

- David Parnas
- Martin Fowler
- Fred Brooks

In session 3, material on these authors will be presented and your research on these individuals will help your understanding.
Q2. Consider the following UML class diagram.

True False:

T  F  Customers can participate in more than one Loyalty Program
T  F  Burning is a kind of transaction
T  F  A Loyalty Program can be set up without any partners.
T  F  A ProgramPartner must deliver a service
T  F  An instance of Membership can only be created when there is a Customer associated with a Loyalty program

Define the class **LoyaltyAccount** in the object-oriented language of your choice. You do not have to define other classes. Assume they exist.
Q3. Build an Analysis Model

In sessions 1 and 2 we've looked at class diagrams in the context of software design, thinking about how a class diagram converts to actual code. In this exercise, I'd like to take a step back and engage in “pure domain analysis” with no concern for object-oriented implementation. Often, at the beginning of a new project that involves a new domain, one of the important jobs of the developer is to “understand” the problem domain. What are the entities and what are their relationships to each other. Once the domain is understood, then there is some hope that the developers will understand what the customer wants. In the early days of OO, the pure domain analysis served as the stepping stone to formulating a design, but had no value other than giving the developers an idea of what the domain was all about. Now, however, that has changed and doing pure analysis of a domain can be captured and with the help of knowledge engineering tools, to add value to an enterprise. Recall Tim O'Reilly's comments about how possessing the data has become the key to the success of many Web 2.0 companies.

In this exercise, use the noun-verb technique to jump start the creation of an analysis model with ONLY class names and general associations (no composition or aggregation, no attributes, no operations). The purpose of the analysis model is to describe a domain only using class names (entities) and associations. Do not think Java or C++ classes – think things or entities in the domain. The end result should be a diagram – a set of boxes and lines that capture some knowledge about the domain. In simple sentences, nouns are good candidates for entities and verbs are good candidates for associations between those entities.

Let's start with a sentence from the SAP website:

SAP was started by five engineers from IBM.

I have made the nouns bold and underlined the relationship. Using the nouns and verbs as a starting point, there are actually two relationships here:

- SAP started-by 5 engineers
- 5 Engineers were employed by IBM

This leads to the following class diagram. Note that the attribute and operations sections are blank and there are roles attached to the relationship to give more semantic information to the viewer.
Let's try another sentence from the web site.

The **Company** lists more than 88,000 **software installations** for over 26,000 **customers** in more than 120 **countries**.

Here there are several nouns and one verb, lists, which is not very informative. After some rewriting I came up with the following associations:

- Sells-to
- Installs
- Sells-in
Identifying *is-a* relationships.

After coming up with entities and relationship taken directly from the text, one can ask “are there any is-a” relationships that might help others understand the domain. Just from knowledge of the world, one can say:

- SAP and IBM are companies
- SAP customers are companies (may be exceptions but I doubt it)
- Engineers are Persons (mostly)

Here we can use the inheritance arrow to indicate the *is-a* relationships.

With only two sentences, we have the start of a domain model for the role of SAP in the software sales world. Note that we are using only:

- Class names (no attributes or operations)
- Inheritance arrow to indicate *is-a*
- Association links and labels to indicate the relationships between entities

How is this useful?

A. If you were tasked with developing software that dealt with the computer industry, this analysis model could be used to help identify the actual Java or C++ classes that will be part of the design. To do this, of course, one needs requirements and use cases.
From a knowledge engineering standpoint, these entities and relationship are a kind of knowledge base or ontology for the software industry. With tools we will look at in this class and in the upcoming XML class, there are ways to store this information in a database and do searches over the data. Remember what Tim O’Reilly says: data is next killer app.

**DO:** Using the above as an example, read the following article from Barron’s about Google and Facebook that relates to Tim O’Reilly presentation we watched (thanks to xx). An interesting aspect of the talk was his insight about the movement of profits from the hardware sector to the OS to the application market to Internet platforms involve users. Note how this theme is appears in the article.

Select at least two sentences from the following article and use the above noun/verb analysis technique to build your own analysis model based on entities (classes) and their relationships (associations) to other entities. For each association that you come up with, give the association a name and add roles that describe the roles of the entities.

Use a UML tool to do this. The above diagrams were done fairly painlessly with ArgoUML. FYI, Snagit was used to capture the diagrams; ArgoUML has an export graphics function.
Google Rides to the Rescue of Facebook's Rivals
By ERIC J. SAVITZ

MARK ZUCKERBERG, THE SANDAL-WEARING 23-year-old paper billionaire who runs Facebook.com, last week found himself in a strange position: back on his heels. The reason hints at some potentially monumental changes now rolling through the Valley.

Facebook, as you certainly know by now, is a white-hot social-networking site, dominating the sector along with MySpace. (MySpace is a unit of News Corp. -- ticker: NWS -- which is in the process of purchasing Barron's publisher Dow Jones.)

What's happened recently is that Microsoft (MSFT) agreed to pay $240 million for a small stake in Facebook, giving the still-private company a seemingly ridiculous $15 billion valuation. To win the deal, Microsoft had to beat out competing bids from Google (GOOG) and Yahoo! (YHOO).

The competition was for a foothold in the potentially lucrative market of serving up ads on the increasingly popular site. Microsoft had the upper hand, having already won the right to serve all the advertising to Facebook's domestic customers; with the new investment, they added the right to serve the ads to Facebook's international customers. (One source told me last week that Facebook had bids from all three companies based on the same $15 billion valuation, but Microsoft was able to win the deal by promising higher revenue-per-thousand for advertising on the site.)

It's important to remember that the version of Facebook that everyone in Silicon Valley is so fascinated with is a lot different from the original version, which functioned as a social-networking tool strictly for college students.

The two big changes: It opened up access to all comers, rather than serving just undergrads, and it opened up the site to third-party applications. The result has been a huge flood of new users and a zillion Facebook applications, many of the dimwitted variety: Throw sheep at your friends! Turn your pals into virtual zombies! Give your buddies a poke! Or better, a super poke! A growing number of companies focus their software development on creating widgets to run on Facebook, the way previous generations of developers wrote programs for Windows.

This rapid success on the part of Facebook in attracting developers -- in addition to users and advertisers -- has caused a complete rethinking of just what social-networking environments are: Rather than simply collections of Websites, they're now being viewed as "platforms."

Facebook and MySpace are the best-known social networks, but there are many more: Friendster, LinkedIn, Bebo and Google's own Orkut, to name a few. But the managements of these smaller players in recent months faced a huge problem: With developers increasingly focused on creating applications for Facebook, there was a risk it would become an eBay-style, winner-take-all behemoth, squashing smaller players.
GOOGLE CAME TO THE RESCUE last week, launching a new standard for developing social-networking applications. Known as OpenSocial, the standard offers developers a write-once approach to creating programs to run on multiple social-networking sites. And more than a dozen social-networking companies, most notably MySpace, vowed to adopt it. Just a few weeks ago, at the Web 2.0 conference in San Francisco, MySpace CEO Chris DeWolfe announced plans to open his site to third-party developers. Then last week, in a joint press conference at the Googleplex in Mountain View with Google CEO Eric Schmidt, DeWolfe announced that OpenSocial would be the standard for MySpace development.

The idea here is that MySpace, when combined with other sites that have signed on to use the standard (like LinkedIn, Orkut, Bebo and others), will provide a target development platform as alluring as Facebook's.

Think of it as shifting from a world in which every computer company has a unique operating system, to one in which everyone can build on the same OS. So OpenSocial could indeed become the Windows of social-networking.

"The biggest thing about this is, it is really good for users," asserts Marc Andreessen, the legendary engineer and entrepreneur behind Netscape, and more recently the founder of Ning, a startup that provides software and services that allow anyone to create a social-networking site. "It gives users more power, more ability to do the things they want to do." And for developers, he says, it offers a standard way to create applications.

So far, Facebook hasn't said whether it will adopt the new standard; Andreessen thinks it doesn't have to, contending it isn't hard to develop applications for either standard. "Macs versus Windows is a fundamental decision," he says. "This is totally different. This is about what happens on the front end. Ninety-nine percent of the effort is on the back end."

But it does change the landscape dramatically, in particular raising questions about Facebook's ability to maintain its dominant position. If the company were public, its stock would have dropped last week. "Facebook was hoping for a proprietary platform, a lock-in like Microsoft has with Windows," Andreessen says, "where applications can only run on Facebook. Now the opportunity for platform lock-in is gone."

Andreessen thinks social-networking is just getting started. He says no one really knows how many active users there are; the number he uses is 200 million, though he concedes it may be half that many. That leaves more than a billion Internet users who aren't using any social networks yet. "It's like the Internet in 1994," Andreessen recalls. "Some people were online, but most were not." The OpenSocial platform, he says, "unlocks the ability to have a million social networks."

Andreessen also notes that Facebook is widely expected to be preparing its own ad network, using data in its user profiles to sell advertisements on third-party sites around the Web in competition with Microsoft, Google and Yahoo!