Gabrielle Sudik
IT1025 CRN 12115 – Spring 2017
Professor Tamerlano
Lab 7
May 1, 2017

Executive Summary

The main topic of this week's lab is systems development, particularly the various ways to develop software, when each approach should be used, and balancing cost, time and quality. The lab also discusses programming languages and web design, and professional organizations working to make software better, safer and more secure.

Systems Development and the Quality Triangle

Systems-development life cycle (SDLC) is a method for managing large projects that include mulitple programmers and systems, and that will have a large impact on a business. It typically consists of seven steps: Preliminary analysis, system analysis, system design, programming, testing, implementation and maintenance. It is intended that each step be finished before the next step begins, and is meant to be very structured and make sure each part is fully understood and working before the next part begins. SDLC can be rigid, slow and resource-consuming.

Rapid application development (RAD) is intended to be a faster implementation system, and overlaps steps of program development. Typically, RAD includes requirements planning, user design, construction and cutover. In theory, RAD encompasses all of the steps of SDLC, but can overlap the steps or have to go back to earlier steps. For example, a RAD system might be made available for users, but the users become the de facto testers of the system, and the programmers might then have to go back to do more programming.

An agile development system make small incremental changes to a system, focusing on quality and attention to detail of each part. Like a RAD system, each small part can be done quickly, and with overlapping steps. But because developers focus on one part at a time, they can ensure a quality end product for each piece, moving on to something new when an earlier component is complete.

A lean development system focuses on building a “minimally viable product” – that is, a product that does the barest minimum required to demonstrate what it can do – then it is shown to users for feedback. This cycle will repeat itself until everyone agrees the final product has been achieved. This development method is used to start creating programs before the users know everything they want the program to do. It is often used to “test drive” a program idea to see if it is worth pursuing in full.

If I were developing a a system, the method I chose would be highly dependent on the circumstances of why I'm developing a program and its ultimate goal. For example, if I were designing a new videogame, I would want to start from scratch and not release the game until it was in perfect working order. Doing otherwise leads to nothing but angry fans and bad reviews. Hence, the SDLC method would be best. It would take a long time, but would produce the best final product.

If I were, say, working for a business that needed to create an internal program quickly, and was willing to forgo a perfect system for one that is better than no system at all, RAD or an agile method might suffice. Both allow for the development of the program while it is getting feedback from users. Users would have to work with an imperfect system for a while, but eventually would acquire a completed system.

An agile method would probably work best if I needed to update or alter an existing system. Users can continue to use the old system, but each week or so they will discover that a new or updated part of the program has been added. This will help continuity of work, and will allow users to learn the new components at a steady pace, rather than being introduced to an entirely new system at one time.

The Quality Triangle says that while developing a systems, creators can focus on only two of the following three limitations: time, cost and quality. In practice this means developers can create a high quality product with a lot of money but in little time, or with a lot of time but little money. If both money and time are short, you will end up with a low quality product.

Assuming that the project I'm working on is SDLC, a high quality product is a must, so it will either take a lot of time or a lot of money or (realistically) both. Time is probably the most important – that is, it will take a long time – because each step needs to be complete and well-done before moving on to the next step.

Other approaches will allow for different outcomes. For example, a RAD approach might be completed quickly (little time) but may cost a lot as several people will need to work on it to get it done. An agile system on the other hand may take a long time, but may be done more cheaply, as only one component at a time needs to be worked on.

Programming Languages

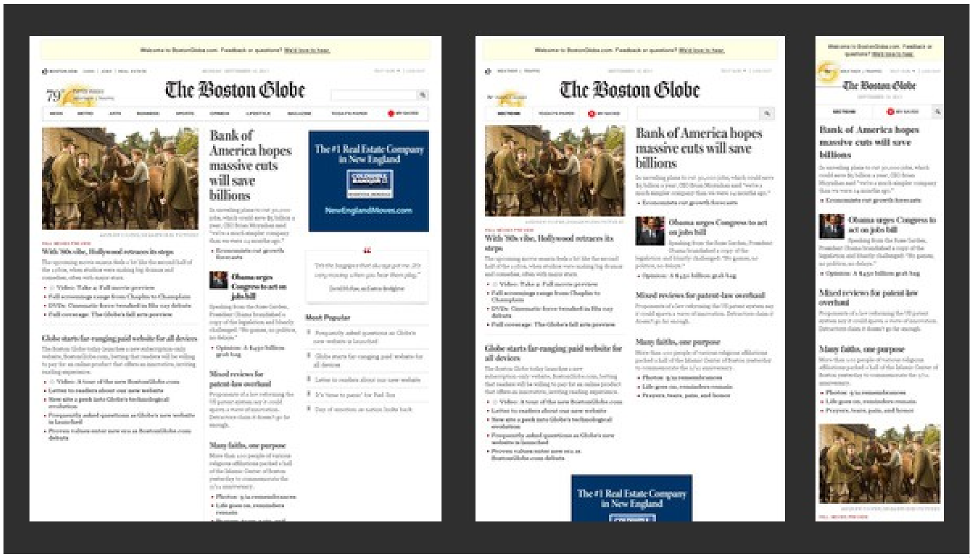
A compiled programming language is when a programming code is translated into a machine-readable code called an executable that runs on hardware. Examples include C, C++ and COBOL. An interpreted language is one that needs a runtime program to execute it. The runtime program then interprets the code and runs it. Examples include BASIC, PHP, PERL and Python.

Javascript is the language that tells web pages how to behave. It is interpreted, object-oriented and third generation. C++ is used to created software of all kinds. It is compiled, object-oriented and third generation. Ruby is also a general-purpose programming language. Like C++ is it object-oriented and third generation, but depending on its implementation, it can be either compiled or interpreted.

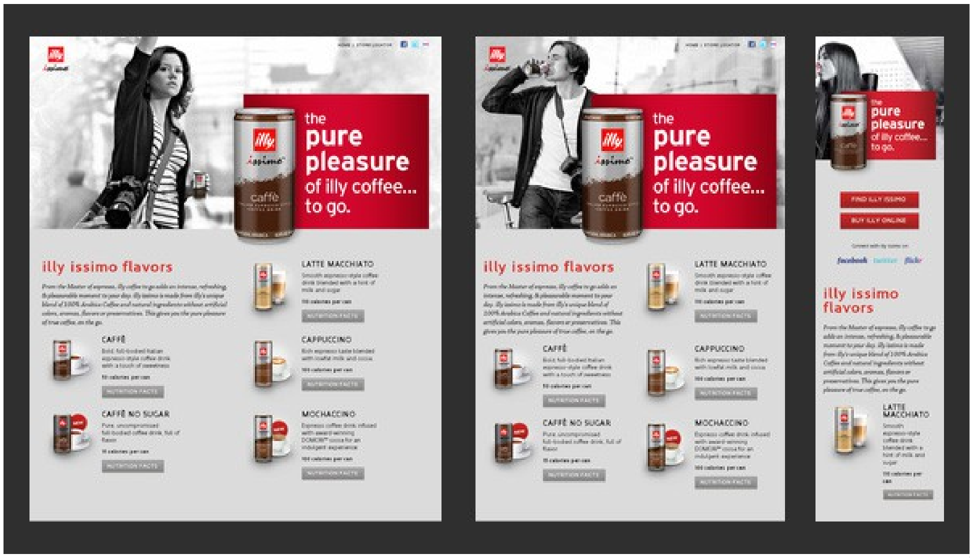
Responsive Web Design

The goal of responsive design is to create a website that works equally well on various devices – computers, phones and portable devices, with various operating systems, screen sizes, and interfaces such as mouse versus touchscreen. Making one site or app that works on multiple devices is far more efficient than trying to create various programs or sites or apps. In addition, the majority of users prefer to browse most web content in a web browser (even on a phone or small tablet) than in a stand-alone app. Three examples of well-done responsive design are posted below, showing how photos change or disappear based on size of screen, and how text columns rearrange. (Screenshots courtesy of designmodo.com):

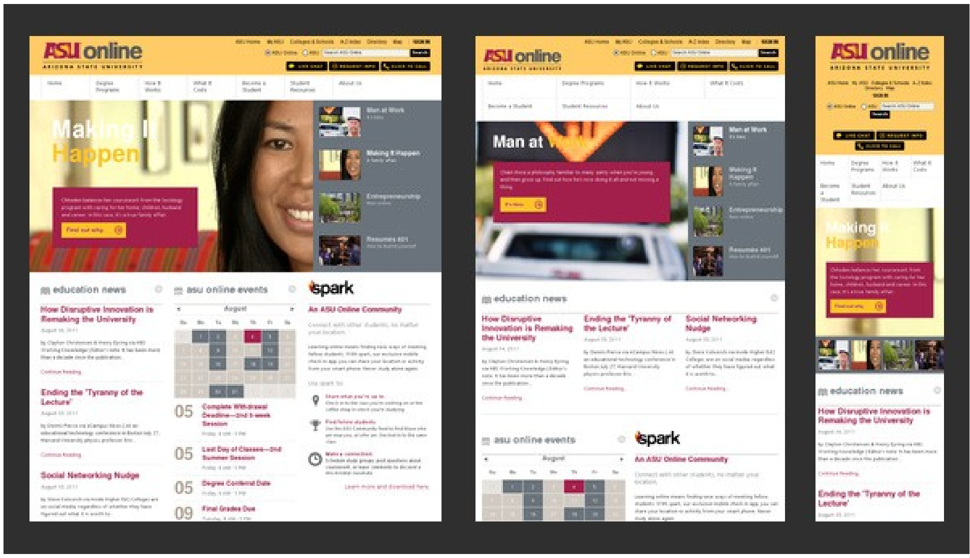
The Boston Globe:


The Boston Globe
Bank of America hopes massive cuts will save billions
The #1 Real Estate Company in New England
NewEnglandMoves.com
Obama urges Congress to act on jobs bill
Mixed reviews for patent-law overhaul
Many faiths, one purpose
Globe starts far-ranging paid website for all devices
Most Popular

Illy Issimo:

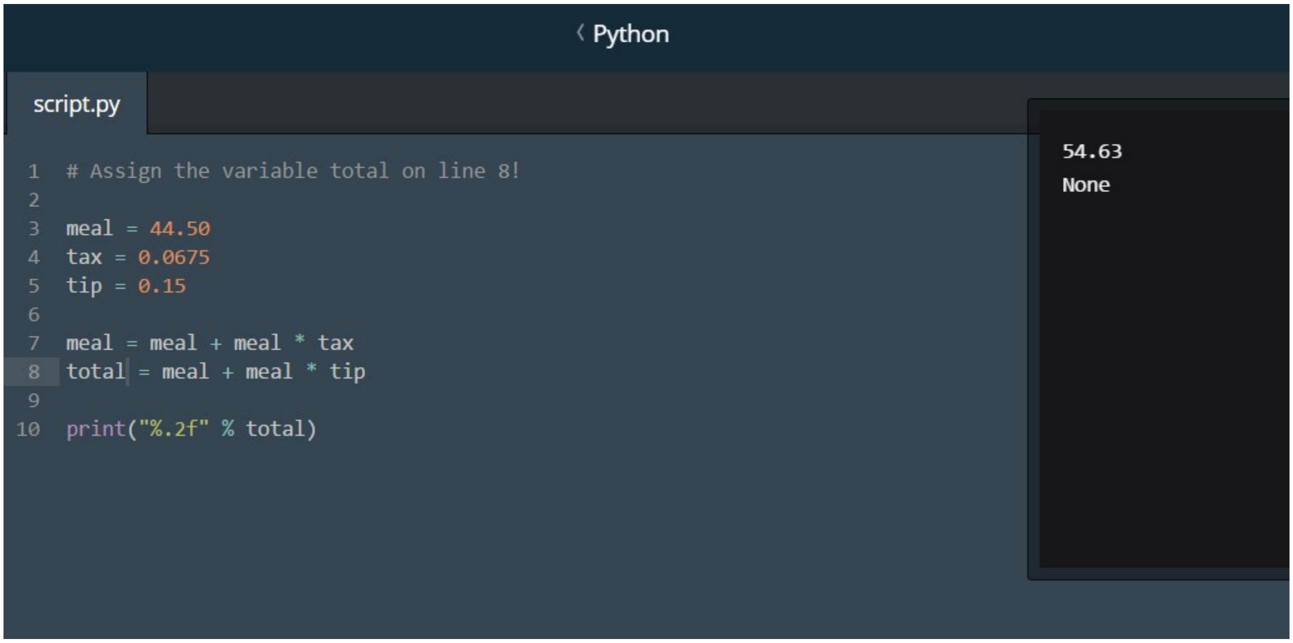


Arizona State University:



Python

I worked my way through CodeAcademy's Python Syntax and Tip Calculator exercises. For the most part they were easy, so long as I typed exactly what the instructions told me. Once or twice I had to figure out something on my own (like what the “divide by” symbol was) but mostly it was a straightforward introduction to Python. I'm sure that memorizing instructions and doing more complex activities will be harder. Here is my final result of calculating the total cost of a meal:



```
1  # Assign the variable total on line 8!
2
3  meal = 44.50
4  tax = 0.0675
5  tip = 0.15
6
7  meal = meal + meal * tax
8  total = meal + meal * tip
9
10 print("%.2f" % total)
```

```
54.63
None
```

[I attempted to do the exercise in Visual Studio, too, but I either downloaded the wrong components or I did not understand the instructions, because I could not even get started with the right part of the program. Perhaps I'll try for fun sometime.]

Object-Oriented Programming and Browser Objects

Object-oriented programming is a design philosophy that groups everything into "objects" that can perform a variety of actions. Basically, a programmer creates objects that are capable of doing things on their own. Each object is then programmed into a larger piece of a program, and each time it is needed, its functions are already defined. Structured programming is a top-down design model, in which developers map out the overall program structure into separate subsections. At least one source on the internet says that OOP is a subset of structured programming.
Examples of OOP languages include Python, Visual Basic, Java, C++ and COBOL (so OOP is not a new concept). Examples of structured programming languages include Pascal and COBOL (implying that at least one OOP language is also a structured language).

There are several browser objects – one of them is the history object, which contains a list of all URLs viewed in a particular browser window. This is a fairly simple object. It can be set to show a certain number of viewed URLs. And it has three different instructions: going to the next URL on the list, going to the previous URL on the list, and going to a specific URL on the list. (Other browser objects are the window, navigator, screen and location.)

Computer Professionals for Social Responsibility

Computer Professionals for Social Responsibility promotes the responsible use of computer technology. Among other things, the CPSR works to foster responsible use of technology, educate the public about its limitations – including the assumption that technology alone can fix all problems, examine social issues in the technology profession, and encourage IT as a way to improve life.

I read CPSR's articles about a data breach at the U.S. Department of Veterans' Affairs, which happened in 2006 when a VA employee's work laptop was stolen from his home, compromising the privacy of 26 million veterans' records. CPSR did not really make this a "project" but did write about the numerous VA errors that allowed the data breach to happen – presumably as a warning to other agencies or companies about how *not* to protect client privacy.

Among other things, CPSR noted there was no encryption of the sensitive data. It's not clear whether the VA had any rules about who could take work home, and what information they could take. There were no rules for how the theft should have been reported to the VA (ie, to make sure that the correct people knew about it).

CERT

CERT is a Division of the Software Engineering Institute, and works to reduce vulnerabilities in software by preventing coding errors, or discovering security flaws during testing. Its work extends beyond just writing code in a secure way; CERT also studies where threats come from and how threats can be reduced or caught. For example, CERT operates an Insider Threat Center, which researches iemployee cybersecurity attacks from both a technical and behavioral point of view. It works with several government agencies, academia and businesses to analyze cases, formulate best practices and identify threats and attacks. Each year, in conjunction with the federal government, CERT publishes the Cybersecurity Watch Survey, to identify electronic crime-fighting developments and techniques, including best practices and emerging trends.

Conclusion

This lab primarily addresses systems development and the related topics of programming languages and web design. It also touched on two organizations working to make computer software better, safer and more secure.