Programming Languages - Writing Project

You are to write a short chapter on some aspect of program language design or implementation that is of interest to you. The audience for your chapter is CS 222 students. I should be able to assign your chapter to future classes and not hear any complaints that the reading was difficult, confusing, trivial, etc. Projects like this can lead to an interesting IS topic!

Chapters can be written individually or in a team of at most three students. The expected length for a one-person team is 2400 words, two-person team is 3600 words, and three-person team is 4800 words. The project will be completed in several stages.

Project Timetable

Below is an encapsulated timetable for project milestones followed by an expanded version, complete with descriptions. All work must be written in LaTeX, submitted as a PDF, and should be error free. See the LaTeX Wiki for a template.

	Assignment	%
Friday, September 11	Team, Topic, & Sources	5
Friday, September 25	Revised Description	10
Friday, October 9	Annotated Bibliography	10
Monday, October 19	Elevator Speeches begin	10
Wednesday, October 28	Draft Chapter	15
Wednesday, November 4	Peer Review	10
Wednesday, November 18	Final Chapter	20
Wednesday, November 18 (+ following days)	Presentations	20

Team, Topic, & Sources

Let me know if you are working individually or with someone. Provide a tentative title for your chapter accompanied by a 120-word description of what the chapter covers and a minimum of six references (use ACM style) you might use as source material for your chapter. Web references and chapters from textbooks are allowed. I will provide feedback from which you will write a revised project description. If you have trouble thinking of a topic, a list of topic ideas is provided at the end of this document.

Revised Description

A revision of your proposal based on my comments. By this point you should have addressed all my questions and have a description of a chapter that can be completed by the due date.

Annotated Bibliography

There should be at least six sources and each team member should have completed at least three of the annotated entries. Indicate who completed which entries. Provide the title for your chapter at the top of the bibliography followed by name(s) of the chapter author(s), a paragraph describing your topic, then the annotated entries.

Elevator Speeches

Imagine you are in an elevator with a textbook publisher. You have ninety seconds (it is a slow elevator) to convince her that your chapter should be included in an upcoming edited volume in issues in programming languages and translators. In class, I will call randomly on one student in each team to give an elevator speech on what you are doing. Periodically, I will call randomly on students to give elevator speeches about their chapter until everyone in the class has given at least one. Being able to give a good elevator speech is important to your career.

Draft Chapter

A draft of your chapter. At least half of the chapter (preferably much more) must be written and there must be equal contributions by all members of the team. Also provide a schedule and division of labor for all remaining work.

Peer Review

A peer review of the assigned chapter draft. You should answer all questions in the review document and provide serious and thoughtful feedback to the author.

Final Chapter

The final version of your chapter, complete with bibliography. Also provide an accounting of the contribution of each member of the team.

Presentations

The presentation should be 10 minutes. As with your chapter, your audience is a class of CS 222 students. Keep in mind that your audience will not have read the materials that you used to write your chapter.

Project Grading

The chapters will be graded based on the following criteria:

- meeting deadlines
- the quality of work done in each step
- final chapter – clarity, cohesiveness, organization, use of examples and illustrations
- presentation

Except in exceptional circumstances that are brought to my attention well before submitting the final version of the chapter, all members of a project team will receive the same grade.

Project Ideas

Pick an area in which you are interested. Do a bit of reading on the subject. I can provide some initial leads but you should do most of the work in tracking relevant sources.

- exception mechanisms – how do different languages support the generation and handling of exceptions? How are these differences important?
- finalizers / destructors – languages such as C++ and Java (also C#?) have the notion of a finalizer, a method that is called when the object is to be removed from memory. Explore what these do for different languages. What issues are there for the programmer to consider?
- software watermarking and obfuscation – this would have to include examples and analysis of algorithms.
- languages that support parameterized (generic) types – how do different languages support generic types? How easy are the features to use? Are there runtime issues? How does genericity work in the face of inheritance, assignment, parameter passing, method return types?
- optimizing virtual machines: HP Dynamo, Transmeta Crusoe, IBM Jikes
- programming language support for concurrency – what support is there for concurrency? What models for concurrency are there?
- garbage collection – what kinds of garbage collection is there? How do they compare in terms of efficiency and complexity? Can garbage collection be done in real time? What are the programmer implications?
- parallelizing compilers
- optimization strategies
- real time programming languages and extensions
- Microsoft's .NET architecture – this would have to include a discussion of managed code, support for multi-language applications, and extensive examples of your own creation.
- inheritance – look at the semantics of inheritance and how it is implemented in different languages. A good place to start is "On the Notion of Inheritance" by Antero Taivalsaari, ACM Computing Surveys, Vol. 28, No. 3, September 1996, pp. 438 – 479.
- languages with multiple inheritance – Java has multiple inheritance of specifications and C++ has also multiple inheritance of implementation. Explore the implications. A starting point might be Cardelli's paper "A Semantics of Multiple Inheritance" in Lecture Notes in Computer Science, vol 173, 1984, pp 51-67.
- comparison of features of functional and imperative languages
- tutorial of a programming language not covered – history, influences, description of language features, examples