

## CIS 455: Internet and Web Systems

<b>Course Number &amp; Title (A.1)</b>	<b>CIS 455 Internet and Web Systems</b>
<b>Credit Units (A.2)</b>	1 CU (3 hours of lecture per week)
<b>Instructor (A.3)</b>	Andreas Haeberlen, Assistant Professor, ahae@cis.upenn.edu, 560 Levine Hall, 215-746-6167
<b>Text(s)/Required Materials (A.4)</b>	Tanenbaum/van Steen, "Distributed Systems: Principles and Paradigms" Additional materials as handouts, e.g., research papers
<b>Catalog Description (A.5a)</b>	<p>This course focuses on the issues encountered in building Internet and web systems: scalability, interoperability (of data and code), atomicity and consistency models, replication, and location of resources, services, and data. Note that it is <i>not</i> about building database-backed or PHP/JSP/Servlet-based web sites (for this, see CIS 330/550 or MKSE 212). Here, we will learn how a Servlet server <i>itself</i> is built!</p> <p>We will examine how XML standards enable information exchange; how web services support cross-platform interoperability (and what their limitations are); how "cloud computing" services work; how to do replication and Akamai-like content distribution; and how application servers provide transaction support in distributed environments. We will study techniques for locating machines, resources, and data (including directory systems, information retrieval indexing and ranking, web search, and publish/subscribe systems); we'll discuss collaborative filtering and mining the Web for patterns; we'll investigate how different architectures support scalability (and the issues they face). We'll also examine the ideas that have been proposed for tomorrow's Web, including the "Semantic Web," and see some of the challenges, research directions, and potential pitfalls.</p> <p>An important goal of the course is not simply to discuss issues and solutions, but to provide hands-on experience with a substantial implementation project. This semester's project will be a peer-to-peer implementation of a Google-style search engine, including distributed, scalable crawling; indexing with ranking; and even PageRank. We will also incorporate the use of topic-specific recognizers and mash-ups in order to support forwarding of certain searches to Google Maps.</p> <p>As a side effect of the material of this course, you will learn about some aspects of large-scale software development: assimilating large APIs, thinking about modularity, reading other people's code, managing versions, debugging, and so on.</p>
<b>Prerequisites (A.5b)</b>	Knowledge of Java programming; CIS 121 + CIS 380 recommended
<b>Course Satisfies (A.5c)</b>	[ ] Math [ ] Science [ ] Engineering [ x ] Technical Elective [ ] TBS ( <b>check only one, UG curric impact only</b> ) <b>Elective (in a group of courses that the students must choose from)</b>
<b>Course Web</b>	<a href="http://www.cis.upenn.edu/~ahaе/teaching/cis455-s11/">http://www.cis.upenn.edu/~ahaе/teaching/cis455-s11/</a>
<b>Course Outcomes (A.6a)</b>	<ul style="list-style-type: none"> <li>• Ability to design highly scalable systems</li> <li>• Ability to implement large and complex software systems - specifically, a multithreaded web server and servlet container, a web crawler, a DHT-based caching system, and a Google-style search engine</li> <li>• Ability to experimentally evaluate the performance of a complex software system</li> <li>• Ability to work in small teams to build a large and complex software system</li> <li>• Ability to identify security problems in Web systems and to apply suitable countermeasures</li> <li>• Understanding of, and ability to work with, a commercial cloud-computing platform</li> <li>• Ability to use unit tests, to debug large software systems, to integrate code from multiple sources, to work with specifications and third-party code, to use different data exchange formats, to program scalable tasks in MapReduce, to work with a variety of Internet and Web technologies, to write and debug concurrent programs, and to build modular systems and decentralized systems</li> <li>• Understanding of algorithmic principles behind decentralized systems and their scalability, strict/loose consistency models and their tradeoffs, indexing structures for large-scale search, fundamentals of information retrieval, techniques for achieving robustness to various types of faults</li> </ul>
<b>Contribution towards Program Outcomes (A.6b)</b>	B, C, D, G, I, J, K
<b>Topics Covered (A.7)</b>	<ul style="list-style-type: none"> <li>• Server architectures: Web/application servers, client/server, P2P, threads, locks, event-driven prog.</li> <li>• Web technologies: Servlets, CGI, HTTP, HTML, DNS, Cookies, ...</li> <li>• Naming &amp; locating resources: Directories, URLs, search, B+ tree, content-based addressing, pub/sub</li> <li>• Representing data: Schemas, XML, XPath, XSLT, DTD, DOM, collaborative filtering</li> <li>• Decentralized systems: Key-based routing, consistent hashing, partitioning, BitTorrent, Chord, Pastry</li> <li>• Storing/distributing/retrieving/processing data: Cloud file systems, MapReduce, Hadoop, Mercator</li> <li>• Code interoperability: RPC, SOAP, WSDL, REST, web services, service composition</li> <li>• Documents and ranking: Information retrieval basics, ranking, web crawlers, HITS, PageRank</li> <li>• Transactions: ACID properties, two-phase commit, application server and TP monitor architectures</li> <li>• Fault tolerance: Replicated state machines, consensus, Paxos, rational behavior, Byzantine faults</li> <li>• Security: Web security, views, ACLs, capabilities, cryptography basics, Kerberos, TLS</li> <li>• Special topics: Accountability, differential privacy</li> </ul>
<b>Grading Details</b>	25% Homework, 15% Midterm 1, 15% Midterm 2, 40% Final project, 5% Participation
<b>Prepared By/Date</b>	Andreas Haeberlen, January 2011