"Polyglot and Polyparadigm Programming"

Speaker:
Dr. Dean Wampler

Abstract:
Is one language and one modularity "paradigm" right for your entire application? Probably not. This talk argues that modern applications are easier to implement and evolve when they combine several, appropriately-chosen programming languages and paradigms (object-oriented, aspect-oriented, functional, etc.).

You probably use any combination of Java, C#, C++, XML, HTML, Javascript, SQL, ant, maven, make, rake, "shell" scripts, etc. in your daily work. You may approach design from an object-oriented or functional perspective. This talk discusses successful examples where different languages and paradigms were combined to create robust and modular applications that evolve gracefully to meet changing requirements. One such example is Emacs. We'll discuss an architectural style I call "components + scripts = applications", where higher-level "policy" code, written in high-productivity scripting languages (e.g., Lisp, Ruby, or JavaScript), is combined with lower-level "components", written for performance or to bridge to 3rd-party and operating-system APIs. We'll also discuss domain-specific languages (DSLs) from this perspective.

Are objects the best way to modularize code? We'll discuss how functional programming in languages like Erlang, Clojure, and Scala makes it easier to write robust concurrent programs. We'll see how problems of "cross-cutting concerns" led to aspect-oriented programming.

Bio Sketch:

Sponsored by Loyola University Chicago’s Department of Computer Science and the Chicago ACM Chapter
Optional RSVP at http://www.chicagoacm.org/rsvp
For more information, see http://www.cs.luc.edu (Upcoming Events)