Contribution ID: **38** Type: **not specified**

Bebop Protocols

Tuesday 10 July 2007 09:30 (30 minutes)

Zope3 has been criticized as an overly complex framework with a steep learning curve. Especially the ZCML configuration language and the missing Python API for configuration actions seems to be an obstacle for Python programmers.

The talk introduces bebop.protocol, an experimental package that tries to combine the conciseness of Python with the explicitness, fine-grained configurability, and conflict management of ZCML. A protocol is a Python class that defines how a component is configured, registered, called, and unregistered. Protocols are used and extended by declarations, i.e. class advisors and decorators that correspond to existing ZCML directives. All declarations within a package can be activated with a single line of ZCML. The equivalent ZCML configuration can be recorded for documentary purposes and used as a basis for more selective configurations and overrides.

In comparison to Grok, which tries to simplify Zope3 by using conventions instead of ZCML, Bebop favors a less radical approach. Grok smashes ZCML, Bebop generates ZCML. Since the protocol package mimics the ZCML directives as closely as possible it provides no extra learning curve for the experienced Zope3 programmer. Predefined protocols are available for adapters, utilities, subscribers, pages, and menus. Since protocols are extensible, they can also be used to define generic functions and extend the component architecture with special forms of utilities and adapter lookup. The zope.fssync package is used as an example that illustrates how existing code could benefit from protocols. Relationships to PEP 3124 and other proposals are also discussed.

Bebop Protocols: http://svn.kmrc.de/projects/devel/bebop.protocol/trunk/src/bebop/protocol/README.txt

Author: Dr OESTERMEIER, Uwe (IWM)

Presenter: Dr OESTERMEIER, Uwe (IWM)

Session Classification: Web Related Technologies