BlueSkiesAhead

RobertTansley gave a great presentation at the 2004 user conference, defining the next generation of DSpace (v2.0?) as a chance to implement significant design changes and functionality enhancements. This page is an attempt to collect links to individual thoughts and group discussions on just what the new and improved DSpace might look like, and how it might be implemented. If you've got content that should be linked here, don't be shy.

Background eading

- Martin Fowler explains a specific example of the Inversion of Control pattern called Dependency Injection. An overview and classification of IoC technologies can be found at http://docs.codehaus.org/display/PICO/Inversion+of+Control+Types
- the ChainOfResponsibilityPattern on the Design Patterns wiki.

IRC Discussions

- " DspaceIrc+2004+04+20 ": JimDowning and MikeSimpson talk about code modularity, framework architectures, dynamically unified generation of API code and documentation, and various other semi-related topics, with RobertTansley chiming in at the end.
- " DspaceIrc+2004+04+26 ": JimDowning and MikeSimpson again, talking more about the Chain of esponsibility design pattern, modularity, the GDF project, and a prototype container authorization scheme.
- " DSpaceIrc+2004+04+28 ": JimDowning, MikeSimpson, and RobertTansley, addressing vocabulary issues and modular frameworks.
- " DSpaceIrc+2004+04+30 ": mostly JimDowning and MikeSimpson, further elaborating on vocabulary, plus discussing authentication/authorization and hashing out more specifics on layering and container implementation.
- " DSpaceIrc+2004+05+19 ": MikeSimpson and RobertTansley ramble about various architectural possibilities, and discuss flexibility vs. complexity.

Individuals

- JimDowning: JimsRandomThoughts, WebApplicationFrameworks
- RichardJones: SubmissionSystem,WorkflowSystem, BrowseSystem
- MikeSimpson: MikeSimpsonNspaceMusings, NSpace entries in the CodeClearingHouse, ThreeSpaceDesignProposal

---

Inline Discussion

JimDowning: Thinking about our architectural requirements : -

- Separation of service interface and implementation (modular architecture) : Needed to support the range of configurations expected for DSpace2.
- Dependency Injection : Enforce encapsulation on service implementations (components) by having a container provide all their dependencies (rather than a reference back to a registry mechanism)
- Plugin architecture : Development of different implementations can be divorced from the core development.
- Chain of esponsibility (a la :MikeSimpson:Mike's demonstrator) : Allows multiple implementations of the same interface. With modification allows multiple implementations to handle a request. This will be particularly powerful in situations such as file identification, metadata extraction and conversion.
  Been thinking that it would be cool to be able to just drop jar files into a dspace/plugins directory and pull the configuration out of META-INF, but it would make the chain of responsibility thing tricky in situations where one implementation only is allowed to handle a request (e.g. Authentication). Too much opportunity for it not to be obvious which component is handling which events.