The breaking point came when we introduced Spring* about three iterations into the project. We were following an "agile architecture" approach: keep it minimal and commit to new architecture features only when the cost of avoiding them exceeds the cost of implementing them. That's what Lean Software Development calls "the last responsible moment." Early on, we had only a casual knowledge of Spring, so we chose not to depend on it, though we all expected to need it later.

When we added Spring, the *.jar* file dependency problems were multiplied by configuration file problems. Each deployment configuration needs its own *beans.xml* file, but well over half of the beans would be duplicated between files—a clear violation of the "don't repeat yourself" principle†—and a sure-fire source of defects. Nobody should have to manually synchronize bean definitions in thousand-line XML files. And, besides, isn't a multi-thousand-line XML file a code smell in its own right?

We needed a solution that would let us modularize Spring beans files, manage *.jar* file dependencies, keep libraries close to the code that uses them, and manage the classpath at build time and at runtime.

### ApplicationContext

Learning Spring is like exploring a vast, unfamiliar territory. It's the NetHack of frameworks; they thought of *everything*. Wandering through the javadoc often yields great rewards, and in this case we hit pay dirt when I stumbled across the "application context" class.

The heart of any Spring application is a "bean factory." A bean factory allows objects to be looked up by name, creates them as needed, and injects configurations and references to other beans. In short, it manages Java objects and their configurations. The most commonly used bean factory implementation reads XML files.

An application context extends the bean factory with the crucial ability to make a chain of nested contexts, as in the "Chain of Responsibility" pattern from *Design Patterns* (Gamma et al. 1994).

The `ApplicationContext` object gave us exactly what we needed: a way to break up our beans into multiple files, loading each file into its own application context.

Then we needed a way to set up a chain of application contexts, preferably without using some giant shell script.

* *http://www.springframework.org/*

† See *The Pragmatic Programmer* by Andrew Hunt and David Thomas (Addison-Wesley Professional).