Refactoring in Large Software Projects - Chapter 2
by Stefan Roock and Martin Lippert

This chapter provides some background information on the concept of refactoring and what it is used for. The concept of an emergent design is explained as a design that is constructed incrementally as more is learned about the software behavior and requirements. Continuous and liberal refactoring produces the emergent design and keeps the code from ageing, thus it is preferred to the traditional method of a big upfront design. Refactorings should be conducted to support changing requirements and to eliminate weaknesses in the code. "Weaknesses are present when the existing system structure hampers or even prevents modifications." (p.17) These weaknesses are called "bad smells" and should not be ignored since they degrade the quality of the code over time.

In conducting these refactorings, small refactorings are always better since they reduce the risk associated with the changes by allowing the code to remain in a useable state. Sometimes it is necessary to make the code more complex in order to guarantee it stays in a useable state until the refactor is completed. The authors refer to this as a "detour" with the appropriate analogy: "build a detour first and then tear up the road." (p.20)

Another supporting task that should always accompany a refactor is that of testing. Refactorings rely on automated test cases to check that the existing functionality is not broken after the refactor is complete. The authors discuss two refactoring techniques - code-first and test-first - and show how the test cases should be modified to accommodate the refactor for each technique.

Looking at ISVis with this knowledge is a primary motivation for refactoring the code. We could say a new requirement for ISVis is that it be platform-independent so refactoring the display-related code is to achieve that end. Additionally, ISVis is plagued with bad smells as we have discussed earlier.

```
@book {
  Roock:Chapter2,
  author = {Stefan Roock and Martin Lippert},
  title = {Refactoring in Large Software Projects},
  publisher = {John Wiley and Sons, Ltd.},
  year = {2003}
}
```