



Software Prototyping - Having your PIE and eating it!

Author: Edward Foster

Relative Costs of Errors during Software Development and Maintenance Cycle

| Stage | Relative Cost of Repair |
|-----------------|-------------------------|
| Requirements | 1 |
| Design | 5 |
| Implementation | 10 |
| Unit Test | 20 |
| Acceptance Test | 50 |
| Maintenance | 200 |



Decision Factors

Application complexity and type, customer characteristics...

Generate shared understanding

Enhance existing SAS skills



Knowledge Generation

| Prototyping Methodology Types | | |
|---------------------------------|--|---|
| Prototyping Method | Usage | Objectives of Prototyping Exercise |
| Evolutionary (Open) Prototyping | Prototype built as part of requirements analysis. The prototype is then used and developed in an iterative develop-review cycle to eventually form the finished product. | To deliver a working system to the users. The development starts with the areas which are best understood. |
| Throwaway (Closed) Prototyping | Prototype built as a rough demonstration of a section of the system. The prototype is used as a discussion point to finalise the requirements by ironing out any remaining issues. The prototype is discarded at the end as it will not form part of the final product. The prototype is not necessarily developed using the same software packages or tools as the final product, but instead may use a dedicated prototyping package, such as a 4G language. | To validate or derive the system requirements. The areas which are most poorly understood are used first to create a prototype. |

Software Prototyping

Risk Reduction

Requirements Risks

Prototyping in action...

Project Information Exchange (PIE)

OLE Automation?

DDE?

PROC ACCESS?



SAS/AF AND PROTOTYPING

SAS/AF may seem like a 'blast from the past' for some developers because SAS has moved into other application development areas, incorporating web development in their SAS/Intnet and Apps Dev Studio packages. However, SAS/AF is still a viable solution for many smaller companies where it provides a cost effective development environment. This is certainly true in the pharmaceutical industry where the near ubiquity of SAS in biometrics means it makes sense to create applications that can directly create and access SAS data, with many companies building 'toolboxes' of small SAS/AF tools to do various tasks with clinical data, ranging from data cleaning environments to drug and adverse event coding.

SAS/AF uses Screen Control Language (SCL) code to provide functionality to the user. In many ways SCL can (and perhaps should) be seen as an extension of the SAS programmer's skill set as SCL can be used in a number of useful situations in dataset code as well. This means that investment in training employees to work with SCL and SAS/AF can be seen as worthwhile in a company which specialises in its use of SAS/BASE reporting and statistical elements as well as other related SAS technologies (such as SAS/GRAPH, SAS/STAT, ODS etc).

The ease with which a working SAS/AF frame can be created shows that SAS/AF can be an effective tool for prototyping. Improvements in version 9 like dot notation in SCL code, object data models and the improved OUI for setting and linking attributes make working with AF much easier compared with the 6.12 implementation. The PIE application was developed using SAS/AF with version 8.2. Conversion to use under version 9.1 is the next stage for this application to take advantage of all the new features, such as improved reporting features in ODS and the use of PROC DOCUMENT to generate multiple report formats. We are also looking at the SAS Open Metadata Architecture and the Management Console to see if we can create a company wide metadata store to link all our applications.

AF also allows you to encapsulate functionality in user defined objects and methods. Many companies already use the notion of stores for SAS code, such as generic macros. These stores can be extended to contain these methods to allow for easy retrieval for future prototyping exercises. Re-use of code is a key component of prototyping that relies on rapid development practices. SAS catalogs with appropriate descriptions are ideal for storing libraries of such code.