Experiences with Architecturing the SGF Application Family

Position Paper (short)
for
International Workshop on
Development and Evolution of Software Architectures for Product Families

Andreas Rösel, Michael Wilcke
ABB Forschungszentrum
Speyerer Str. 4
69115 Heidelberg
E-Mail: roesel@decrc.abb.de, wilcke@decrc.abb.de

Introduction

The SGF Application Family consists of a number of internal and external products based on a generic architecture realized as an object oriented framework. It is an advanced object and reuse technology initiative coordinated by ABB Corporate research in cooperation with a number product development units. The systematic use of this technology provides a tremendous benefit for ABB through enabling reuse across projects in various operational units and by introducing the reuse concepts of object oriented technology to the product level. SGF Family applications are now running on about 100 workplaces. (At the 1996 ObjectWorld conference ABB received an Object-Application-Award for the SGF Application Family as 'Best cost saving implementation using an object approach'.) While concrete successes have been achieved there are a number of areas in which research efforts are continuing.

Need for generic SGF architecture within ABB

Within ABB, many Graphical Engineering tools are in use often they are an integral part of complex and application specific processes and environments. Common to all these tools is that they use a graphical representation of a collection of components. In contrast to drawing tools, they usually add some very specific application functionality that is related to the *semantic* of these components. ... To address this need, core requirements for ABB Graphical Engineering tools have been identified, architectured and made available in an object oriented framework that can be used for rapid application development - the ABB Semantic Graphics Framework. Any new graphical engineering application based on the SGF need only add its specifics (that is symbols, their meaning, constraints, special integration interfaces etc.).

Architectural Approach - Layered Frameworks

The architectural approach separates generic aspects from product variants. The architecture is realized as a set of layered frameworks: for example the SGF uses as an underlying base a (single rooted) commercial GUI class library, secondly the SGF base

framework provides the generic application architecture and features common to all applications, at another layer the application specific variants are realized using a combination of black-box and white box techniques. Note that new members to the SGF Application Family may also require extensions which are more generally applicable. These can flow back into the SGF base.

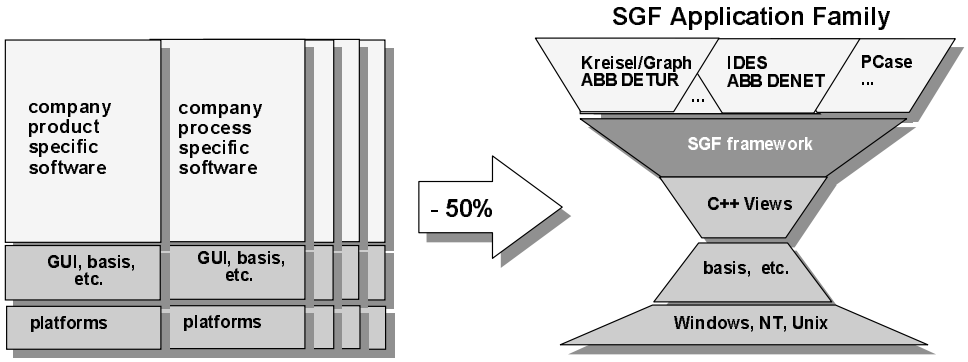

Figure: Layered Framework Approach versus Traditional development

Reuse Benefits

A relative innovative aspect is the conscious layering of object oriented frameworks to maximize their benefit. Each user of a certain layer can benefit from reuse and focus on more specific application competence. The customizability of object oriented technology is utilized for custom specific extension with added value.

The **reuse benefit** is tremendous. Using a simple metrics of classes developed we arrive at reuse savings during development of greater than 50%. These benefits are directly attributable to use of object-oriented framework based development and the layering of these frameworks. For prototypes the major benefit is that they can be developed at all given time and budget constraints. Therefore it is more appropriate to focus on the decision impact rather than on the reuse level (which is very high).

The **maintenance savings have been found to be at least as high as those during development**. ...

Speed of development is another benefit of SGF Applications...

Cost savings ...

Implementation

The SGF Application Family consists of a little more than 1200 classes. ...

Customization of SGF

The ability to extensively customize the SGF base framework is only possible through object oriented technology, this provides the key for a family of SGF applications. Customization takes place in form of configuration (black-box approach) and tailoring (white-box approach).

Configuration

...

Tailoring

...

Example Products of the SGF Application Family

Currently 10 applications and prototypes of various complexity make up the SGF application family. Here brief descriptions of two members follow.

...

Further work / Areas of interest

In the following areas some research results and practical experiences have been achieved. The questions indicate further interest and planned activities:

Development Method / Process

How to best describe the process of systematically developing generic architectures? Transferability between implementation environments (e.g. C++ / Smalltalk / Java)? Coordinating / optimizing interaction of development / maintenance of generic architecture (framework) and products? Scaling up development process: multi-site / larger groups?

Tools

Support for navigating generic architectures (browsers). (Semi-)automatic generation of documentation (e.g. Together/C++ has been used). Tools for testing and consistency checks provided at the base framework layer?

Customizability

Black-box versus white-box trade-off; managing and combining product variants; utilization of standard design patterns; additional domain specific design patterns?

Organization

Adapted organizational structures. Responsibilities for maintenance of generic parts? Cost sharing / Licensing issues?