Experience Report

Implementing Faceted Classification for Software Reuse

Rubén Prieto-Díaz

The Contel Technology Center
15000 Conference Center Drive
Chantilly, VA 22021

ABSTRACT

Experience on the development, implementation, and deployment of reuse library technology is reported in this paper. The focus is on organizing software collections for reuse using faceted classification. Briefly described are the successful GTE Data Services' Asset Management Program and the steps taken at Contel for furthering reuse technology. The technology developed for reuse libraries is presented first, followed by a description of how it was transferred. The conclusions of the experience are: reuse library technology is available, it is transferable, and it definitely has a positive financial impact on the organization implementing it.

1. THE TECHNOLOGY

The central component of the technology reported in this paper is a software reuse library organized around a faceted classification scheme. The system supports search and retrieval of reusable components, and librarian functions such as cataloging and classification. To be effective, the system must operate within the context of an organizational infrastructure aimed at promoting reusability.

1.1 Library Concepts

A fundamental problem in software reuse is organizing collections of reusable components for effective search and retrieval. The concept of faceted classification introduced in [2, 3] presents a partial solution to this problem. Faceted classification offers certain features that not only improve search and retrieval, but also supports the potential reuser's selection process and contributes to the development of a standard vocabulary for software attributes.

In faceted classification the librarian creates a class to match the attributes of the item. It is called "synthetic" because component descriptors are assembled by selecting predefined keywords from faceted lists and it provides higher accuracy and flexibility in classification. In contrast, typical classification schemes such as Library of Congress and Dewey Decimal are enumerative. In an enumerative scheme all possible classes are predefined. The librarian selects the class that best fit the attributes of the new item by traversing the classification hierarchy.

In the Dewey Decimal system [8], for example, the title *Structured Systems Programming* could be classified in any of the following classes: systems analysis (001.61), software (001.642 5), systems (003), systems analysis (from the technology branch, 620.72) or systems construction (620.73). To compensate for such ambiguity cross references are established, but cross referencing is a cumbersome and error-prone process. The faceted approach, in contrast, synthesizes a new class, tailored to the particular title or component. In a faceted scheme developed for Unix components, for example, the class

locate/line-number/file/line-editor

was synthesized using terms from four facets and may only be applicable to one component.

1.2 Why Faceted classification?

Current approaches to information retrieval are often classified along a scale whose end points are free text analysis and controlled vocabulary. Free text analysis, also referred to as uncontrolled vocabulary, consists in analyzing word frequencies in natural text [5]. Relevant keywords are derived automatically by their statistical and positional properties, thus resulting in what is called automatic "indexing," which is the assignment of preferred terms to represent or define an item. A fundamental assumption in this approach is the existence of a large "corpus" of text to justify the statistical analysis. This technique has proven relatively effective for text intensive

0270-5257/90/0000/0300/$01.00 © 1990 IEEE

Recommended by: Herbert Weber

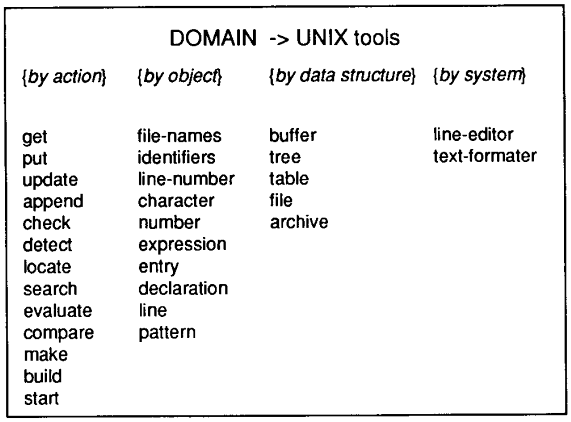
documents such as books and journal articles. Recent reports, however, question the effectiveness of these "purely syntactic" approaches [6].

Controlled vocabulary approaches, on the other hand, rely on a predefined set of keywords used as indexing terms. These keywords are derived and defined by experts and are designed to best describe or represent concepts relevant to the domain of discourse. Software products have certain characteristics that make controlled vocabulary a more attractive approach over free text analysis: 1) source code is very low on free text, 2) keyword meanings are usually assigned by convention or by programmer preference, 3) it is not obvious *what* components do and *how* they do it, and 4) human intervention is typical in extracting meaningful index terms.

In a controlled vocabulary, keyword-represented concepts are organized as a classification scheme. A classification scheme provides a network of predefined relationships thus introducing some semantic information absent in free text analysis.

A classification scheme for collections of reusable software components should meet the following criteria: 1) It must accommodate continually expanding collections, a characteristic of most software organizations, 2) It must support finding components that are similar, not just exact matches, 3) It must support finding functionally equivalent components across domains (both 2 and 3 are requirements for effective reuse), 4) It must be very precise and have high descriptive power (both are necessary conditions for classifying and cataloging software), 5) It must be easy to maintain, that is, add, delete, and update class structure and vocabulary without need to reclassify, 6) It must be easily usable by both the librarian and end user, and 7) It must be amenable to automation.

Faceted classification meets most of these demands. Expansion and maintenance is easily accomplished by adding and updating the faceted lists. Precision and descriptive power is a consequence of the synthetic approach to classification. A consistent list of terms for each facet provides for a standard vocabulary that is common to both the librarian and the user, and its tabular format makes it easy to implement as a relational data base. Therefore, a simplified faceted approach was selected for reusable software, over the classical enumerative schemes. A conceptual graph structure was superimposed on each set of facet terms to measure class similarity and a thesaurus was added to each facet. Thesauri enhance query formulation and classification, support concept clarification and simplify maintenance. Figure 1 below shows a simplified faceted scheme (4 facets) developed for a sample of Unix components.



DOMAIN -> UNIX tools

{by action}	*{by object}*	*{by data structure}*	*{by system}*
get	file-names	buffer	line-editor
put	identifiers	tree	text-formater
update	line-number	table	
append	character	file	
check	number	archive	
detect	expression		
locate	entry		
search	declaration		
evaluate	line		
compare	pattern		
make			
build			
start			

Figure 1. A Simplified Faceted Scheme for Unix Components

A prototype library system was developed to demonstrate all of these concepts. It was part of a research program at University of California, Irvine [3].

1.3 Search and retrieval support system

The first experimental prototype had three major functional components: query formulation, retrieval, and ranking. During query formulation, the reuser creates descriptors of desired components by selecting terms from each facet of the classification scheme. The thesauri are used for context clarification and as a quick and indirect way to define facet terms. A query consisting of valid terms can be used for retrieval, and then modified, making it more general or more specific by adding or deleting descriptor terms. A major feature is query expansion. If the given query returns no hits, the system creates, under reuser request, new descriptors of similar components. The reuser may then select from components that perform similar functions.

The ranking subsystem is based on reuse related metrics. It estimates, for each retrieved candidate, the relative effort it would take to reuse it (i.e., adapt and integrate) in the new system. Objective and subjective absolute metrics are normalized using fuzzy set concepts. These absolute metrics are then made relative to the skill level of the reuser, power of the environment, and several other factors that influence the perception of reuse effort. A reuse effort index is computed and used to rank the list of candidates for reuse.

The system was tested and evaluated for retrieval, classification, and reuse-effort estimation [3]. Retrieval effectiveness was tested by comparing recall and precision values of the prototype system to those of a retrieval system not organized by a classification scheme. The experiments showed a four-fold improvement on precision/recall ratio. The classification scheme was tested for ease of use, accuracy, and consistency. Results were very positive in all three. The ranking subsystem was compared to human ranking for a fixed set of candidates with almost 100% correlation. These encouraging results prompted the development of a second prototype to be deployed in a production environment.

1.4 Librarian support

Several lessons were learned in the first prototype. One was the need to support cataloging and classification of new items and some means of updating the thesauri tables and the classification scheme. A second prototype, developed at GTE Laboratories, included these librarian support functions. A much better interface was also added, as well as several minor features such as a spelling checker, a system usage logging capability, and system usage report generation. These last two features were considered essential to learn how reusers approach the system, what information they look for, and ultimately, what the role of a library system is in the reuse process. This second prototype was tested and used as a model for developing a production system for a real software development environment [1].

1.5 Organizational support

Reusability will not happen by itself, however, by simply building a reusable component library. There must be a strong organizational commitment to reusability and an effective management structure to operate a reusability program with the resources and authority it needs to provide the overall culture to foster reuse. Therefore, an organizational infrastructure is needed for a reuse system to succeed.

The following are the support groups created as part of GTE Data Services' "Asset Management Program" for developing a reuse culture in the organization: 1) first and foremost is a management support group to provide initiatives, funding, and policies for reuse, 2) an accessible, densely populated, fully supported, and easy to use library system, 3) an identification and qualification group responsible for the quality of the repository that identifies potential reusability areas and collects, procures, and certifies new additions to the repository, 4) a maintenance group that maintains and updates reusable components, 5) a development group that creates new reusable components (both 4 and 5 are under the qualification group), and 6) a reuser support group that assists and trains reusers and run tests and evaluations of reusable components [1].

2. TECHNOLOGY TRANSFER

Having the ingredients available, the next step was to make such technology effective. The first technology transfer experience was from GTE Laboratories to GTE Data Services. A more recent effort is underway at Contel Corporation. Both experiences demonstrate the benefits of library supported reuse technology.

2.1 GTE

In 1986 GTE Data Services launched the Asset Management Program (AMP) for the purpose of creating, maintaining, and making available, at the corporate level, a collection of reusable assets. A reusable asset is broadly defined as any facility that can be reused in the process of producing software. Initial emphasis was on reusable software components.

An organizational infrastructure, as described in 1.5 above, was created to support the effort. The Software Reuse Project at GTE Laboratories was given the task of developing the technology: a library system for reusable software.

2.1.1 Implementation Strategy

With the prototype from University of California on hand, a strategy was drafted for an effective technology transfer. Rather than delivering the original proof-of-concept prototype from academia, a joint development program was established. A second prototype (as described in 1.4 above) was developed in parallel but out-of-phase with a production system [4]. This "prototype-slightly-ahead" approach: 1) forced the technology recipient team to learn and understand the theory, 2) encouraged continuous interaction between both the research and development teams, and 3) provided enough margin to test and verify new features and ideas.

The prototype was implemented in an IBM PC environment, coded in C, and used ORACLE as the underlying DBMS. In contrast, the production system was implemented in an IBM mainframe environment, coded in COBOL, and used DB2 as the DBMS. Both systems are functionally and architecturally equivalent. This approach proved very successful and established an effective mechanism to transfer new ideas into the production system.

2.1.2 Classification experience

As the system was deployed in the AMP development environment, a classification bottleneck became evident. Despite the librarian support provided by the system, it was difficult and time consuming to classify new components into the collection. There were two sources to this problem, the incompleteness of the initial vocabulary and the lack of technical librarianship know-how.

A system analyst was briefly introduced to faceted classification and assigned the role of librarian. Unfortunately, the classification scheme and vocabulary were developed by the prototyping team thus requiring a long one-to-one librarian training period. A manual for the librarian was later developed that included not only classification guidelines for new components but a detailed explanation for developing domain specific faceted schemes. The manual was very successful in freeing, at least partially, members of the development team from training duties.

An interesting finding in this experience was that faceted schemes are more effective for domain specific collections than for broad, heterogeneous collections. A faceted scheme for a diversified collection becomes too general, losing its descriptive precision. What became apparent from this experience was the need for tools to support the creation of domain specific faceted schemes and learning that several domain specific libraries are more effective for reuse than a single universal collection.

2.1.3 Usage experience

GTE Data Services' AMP has been very successful. During its first year, with only 38% of the assets in the library being actively reused, a reuse factor of 14% was achieved. The reuse factor was calculated by dividing the lines of code reused by the total lines of code produced by the organization. An estimated $1.5 million overall savings was realized. Figures for 1988 are not yet available, but are believed to show close to a 20% reuse factor. A 50% reuse factor is predicted by the end of the fifth year with a savings of well over $10 million [7].

Although the library system is only one of the components in the AMP, its overall impact has been significant.

2.2 Contel

More recently the Contel Technology Center has initiated a software reuse project. One of the goals of the reuse project is to develop and transfer reuse technology throughout the corporation. Of particular interest are reuse libraries. The natural strategy has been to apply the lessons learned from the previous experiences.

2.2.1 Implementation Strategy

In contrast with the GTE experience, the strategy in Contel is to deploy highly adaptable domain specific library systems that can be customized to organization needs.

Rather than developing a reuse library system for a centralized repository, the strategy has been to develop a generic design that can be instantiated in different environments. This generic design is aimed at supporting domain specific collections and be easily integrated into an existing environment. Tools are being developed to support specialized activities such as deriving faceted schemes, domain analysis, vocabulary maintenance, classification, and component integration. The expected end-product will be a configurable reuse-centered software development environment with the library system as the focal component.

Domain analysis of Contel mainstream software applications is also being done in parallel. Top level generic architectures for each application domain will be used as component integration templates. The reuse environment is expected to support both top-down and bottom-up development. In this approach, a generic system architecture is first configured at a high level to meet specific customer requirements. The library will then be searched to find components to fill the architectural skeleton.

The proposed strategy is expected to substantially improve technology transfer. Manuals ranging from guidelines for coding reusable software, to deriving faceted schemes, to system operation are being written to facilitate technology transfer. A program is being drafted to support technology transfer activities such as training, deriving initial faceted schemes, and consultation on required organizational support.

2.2.2 Classification experience

As a first step in the reuse project, a library system prototype was recently implemented and deployed for observation. A faceted scheme has been derived for the Command and Control domain and a librarian is being trained to expand the scheme and to derive schemes for other domains. In contrast with our GTE experience, the librarian has been in close collaboration with the development team and is fully familiar with both, the application domain and librarianship techniques. Close teamwork has contributed to a draft of specific tool requirements that will support several of the librarian tasks. A second prototype, already in the design board, will include some of these tools.

Our experience in deploying this prototype indicates that the role of the librarian is critical for a successful reuse program. Software collections expand continuously, even in very narrow domains, creating the need to find proper abstractions and to avoid uncontrolled growth. Tools that help the librarian to visualize and understand the abstraction and specialization process during class definition and classification are critical. It is necessary to support the process of discovering classes of components that have the highest reuse potential, and with activities closely related to domain analysis where domain expertise is also necessary.

2.2.3 Usage Experience

Because of its recent delivery (March, 1989) usage experience has been limited to extensive testing and evaluation by the developing team. Also, few software engineers from the recipient organization have been exposed to the library system. Extensive system usage is expected as the current command and control application is delivered, and a new one begins development within the next few months.

A generic library system design is being developed for insertion in other Contel divisions. Work in this system has revealed how important domain analysis is for successful reuse. Component integration and adaptation into a new system is very difficult unless a high level structure is available where components can be plugged in.

3. CONCLUSION

Space limitation in this paper did not allow for treatment of our experience with other very important aspects of reuse such as reuse incentives, organizational infrastructure, the role of domain analysis, and several others. Instead the focus was on classification technology and its deployment to production environments.

As demonstrated by the GTE experience, library based reuse not only works but it makes a substantial financial impact to the organization that implements it. It also has been shown that the technology is available and transferable, providing a transfer plan is followed. Even though our experience has been very fruitful, it is still modest and there is much learning ahead. Preliminary results from current experiments and development at Contel promise a much higher financial impact to the organization and offer the potential for packaging and marketing reuse technology.

Our research and development experience has also pointed to new research directions needed to advance reuse technology. These areas include domain analysis and tools to support domain analysis activities, tool integration for supporting a library centered environment, and most important, empirical evaluation of library systems in an organizational context.

Software reuse is a new field that offers a high payoff in a relatively short term. We should exploit it.

4. REFERENCES

[1] Jones, G. and R. Prieto-Díaz, "Building and Managing Software Libraries." In *COMPSAC88, Proceedings of the 12th Annual International Computer Software & Applications Conference*, pp:228-236, IEEE Computer Society Press, Chicago, IL, October, 1988.

[2] Prieto-Díaz, R. and P. Freeman, "Classifying Software for Reusability," *IEEE Software*, 4(1):6-16, January 1987.

[3] Prieto-Díaz, R. *A Software Classification Scheme*, Ph.D. dissertation, Department of Information and Computer Science, University of California, Irvine, 1985.

[4] Prieto-Díaz, R. and M. Swanson, "Parallel Development: A Case Study in Technology Transfer". In Stan Przybylinski and Pricilla J. Fowler (Eds), *Proceedings of the Workshop on Transferring Software Engineering Tool Technology*, pp:112-113, IEEE Computer Society, Santa Barbara, CA, November, 1987.

[5] Salton, G. and M.J. McGill, *Introduction to Modern Information Retrieval*, McGraw-Hill, New York, 1983.

[6] Salton, G. and M. Smith, "On the Application of Syntactic Methodologies in Automatic Text Analysis." In N.J. Belkin and C.J. van Rijsbergen (Eds), *SIGIR'89, Proceedings of the 12th Annual International ACMSIGIR Conference on R&D in Information Retrieval*, pp:137-150, ACM Press, Cambridge, MA, June, 1989.

[7] Swanson, M.E. and S. Curry, "Results of an Asset Engineering Program Predicting the Impact of Software Reuse." Paper presented at the *Fall Conference on Software Reusability and Maintainability*, The National Institute for Software Quality & Productivity, Bethesda, MD, September 16, 1987.

[8] Dewey, M. *Decimal Classification and Relative Index*, 19th edition, Forest Press Inc., Albany, NY, 1979.