Written by Alan Lawson, May 2003 TA000243APM

Technology Infrastructure

Butler Group Subscription Services

Application Modernisation
TECHNOLOGY AUDIT

Semantic Designs

Design Maintenance System Software Reengineering Toolkit (DMS SRT) 1.1

Summary

Semantic Designs champions the view that software engineers typically maintain code, when a more valuable approach would be maintaining the design concepts, the implementation steps and the relation between the two that led to the creation of the code. The Design Maintenance System Software Reengineering Toolkit (DMS SRT) is capable of automating the capture, analysis and transformation of low-level specifications and implementation steps, focusing on the code and background implementation knowledge required for any particular project. As a tool created to support the Semantic Designs vision, the SRT is capable of automated document extraction, source code pretty-printing, test code coverage and metric extraction tools, as well as strong support for general automated reengineering up to and including application migrations This enables the impact of changes to the software to be predicted and accurately modelled prior to any alterations being made, and greatly reduces the risks of enterprise software projects. The SRT is supported by a rich and extensive list of languages and grammars, and is flexible enough to prove a valuable developer tool for modifying enterprise mainframe applications.

The Semantic Designs standpoint of design maintenance has resulted in a software tool kit that is capable of large-scale software analysis and enhancement in a reliable and cost-effective manner. Although still a small company, Semantic Designs has developed advanced technology and effective software tools that can make significant contributions to the operation of enterprise software projects.

Butler Group believes that the DMS SRT is an excellent example of a product set designed to aid enterprises understand what their core applications are actually doing, and how they are doing so. As part of any evaluation and assessment exercise, tools such as this will play an invaluable role, and the cost-effective nature of the DMS SRT should make its use even more compelling.

The only issue here is that the market is currently fixated upon non-invasive approaches, which commonly claim no need to understand underlying applications – Butler Group's view is that this is a very short-sighted view, and that knowledge regarding core applications must be captured.

Butler Group

MARKET ANALYSIS

The issue faced by organisations that have been operating core elements of their business based upon solutions that have been in place for an extended period of time is that those solutions were never designed to interact in a flexible manner with the rest of the enterprise. Although still performing vital tasks, some enterprise investments are simply out of step with modern business expectations and strategies – logic dictates that a choice must be made between either replacing these systems outright, or finding some means by which their functionality can be accessed in more dynamic, yet still cost-effective, ways. In many cases, and certainly in the immediate future of the Application Modernisation market, the latter option is receiving serious consideration at the enterprise level.

Non-invasive techniques, such as screen scraping, have been used to capture mainframe-based activity and data in order to present them in more user-friendly formats. The current trend in the market is to consider the use of added layers of functionality positioned between end-users and the core business applications, with Web service protocols, particularly eXtensible Markup Language (XML), being used to enable effective exchanges between front- and back-office applications.

Although these Web service layers do create further layers of complexity and additional requirements for system and process management, they also promise greater access to existing functionality without the risk of compromising the core applications that the business is dependent upon. Because Web service protocols pose little additional expense and are also inherently low in risk when deployed internally, their market value is currently compelling.

In direct contrast, invasive procedures and technologies involve direct alteration to the existing enterprise solutions, and tend to be seen as high risk. The applications in question are often poorly documented and understood, making the prospect of effecting change in a safe manner an uncertain one. Long-term, however, the benefits of successfully modernising an enterprise application, for example by transforming business logic into reusable components, or even migrating the solution to a different platform, can be considerable.

The organisation will be able to reduce maintenance costs, through the elimination of redundant code, and closely control application and process activity based on greater levels of understanding of what the system is actually doing. The core activity of the organisation will be future-proofed against new technology developments and business demands, which is not the case with non-invasive alternatives – but the perceived risk of tampering with business-critical systems often proves to be a powerful inhibitor.

The Application Modernisation market is truly horizontal, as companies in every industry sector and of every size are faced by the demand to improve the accessibility of their core applications and ensure their ongoing value to the business. In Butler Group's view, the choice is not whether or not to undertake Application Modernisation, but when and how the process should be done: and although a short-term solution using non-invasive techniques will generate rapid results, the only sure route to long-term future-proofing lies with direct, invasive, work upon the applications identified for change.

▶ OVERVIEW

The Application Modernisation market covers a range of products and approaches, with only the common theme of attempting to extract functionality from existing systems in more effective ways; whether emulation-based, transformational, or Web service-oriented, vendors are attempting to provide opportunities to acquire and utilise business processes based on older technologies with greater flexibility.

The dilemma for modern businesses is that although systems that have supported their efforts for many years are still productive and dependable, these self-same systems do not lend themselves well to modernisation, or integration with agile business strategies. Compounding this issue is the fact that technological advances can quickly relegate even the most contemporary systems to a legacy status.

This latter point reminds us that Application Modernisation is not purely a mainframe issue at all; the drivers toward modernisation begin to bite wherever business processes are undermined by poor interfacing between systems and users, as well as applications themselves.

Semantic Designs approaches modernisation from an interesting perspective. Instead of focusing just upon the maintenance and documentation of code with the goal of reuse, the company instead stresses the capture of the intent of the code, and the reasons for its design and evolution. This notion of design requires the capture of specifications, implementation steps, and justification of the implementation steps. The specification knowledge is a way to encode descriptions about problems of interest, whilst the implementation knowledge is a way to encode engineering “how-to” knowledge. The approach is called the Design Maintenance System (DMS), because Semantic Designs believes that what engineers should be doing is maintaining designs, not merely maintaining code.

In accordance with this viewpoint, Semantic Designs has constructed a set of tools, the DMS Software Reengineering Toolkit (SRT), which focuses on capturing low-level specifications and implementation steps. The SRT provides an organisation with the capability to define and automate arbitrarily complicated static analyses of large software systems, or arbitrarily complicated modifications of large software systems. This provides considerable capabilities in enabling changes to be made to enterprise software, for instance, massive migration of one application language/platform to another, or the analysis of a large system to find dead and redundant code and remove it automatically.

The DMS SRT provides a rich environment for problem domain definition supported by machine-processable semantics, in effect enabling the product to accept and process an engineer’s problem stated in a form suitable to the engineer’s problem domain. Specifications rendered in the domain specific language may be systematically analysed and the analysis results presented in a form suitable to the original problem’s domain-specific language. Finally, by combining DMS support for alternative domain-specific languages with a high performance, semantic-based transformation engine, Semantic Designs delivers a powerful means to support engineers as they move from problem specification through to solution, and onto full lifecycle application maintenance.

Butler Group believes that a wide range of functionality and disciplines will be required in order to extract truly valuable results from enterprise software projects, ranging between the discovery of application and program resource through to the management of new processes based upon components of business logic; every product and service attempting to support such projects should therefore be flexible enough to accommodate several, if not all, of the elements that now come under detailed consideration.

DISCOVERY OF APPLICATIONS

In many cases, the legacy environment will be both complex and poorly documented. If applications are to effectively interact against this backdrop, modernisation tools must first act to inventory legacy systems, determine relationships in order to preserve the business process flow, and establish what code and data elements are present. Ideally, this process should be clearly documented, or alternatively the modernisation tool should capture or create documentation from the legacy source itself.

DMS tools parse 'main' application files in order to discover the auxiliary files that are being referenced, after which application analysis can be carried out to determine data relationships to other application codes. In addition to capturing the data relationships of application sets, solutions dedicated to enabling manageable change to be carried out on enterprise code may also assess the content of individual application programs. The DMS SRT provides a range of options that can be used to build both analysis and modification tools, including:

- Full UNICODE-based parser and lexer generation, including automatic error recovery. Multiple source files can be read, if needed, in multiple source languages. Pre-processing support handles conditionals, macros, and INCLUDE file management as appropriate. Generalised Left-to-Right Parsing (GLR) is the basis of this technology, which can handle any context-free languages.
- Abstract syntax trees can be automatically constructed, and literal values, such as numbers, or escaped strings, can be converted to native, normalised binary values for rapid internal manipulation. In addition, source comments (often crucial to understanding the intent of the original development team) can be captured and attached to AST nodes.
- Pretty-printer generation converts ASTs back to legal source file formats, and can be configured to specify information layouts; this can also include source comments. A fidelity-printing mode can be used to preserve comments, spacing, and lexical formatting of code, without changing the source code. Customised configuration can include the generation of eXtensible Markup Language (XML), HTML, or even obfuscated source text. A particular strength of the DMS SRT is its ability to handle very high volumes of code in order to capture information using the tools mentioned above – millions of lines of code can be assessed accurately in a realistic time frame, as opposed to a manual assessment, which would take man-years of work.

APPLICATION ANALYSIS

Having determined what elements of the legacy system are present through discovery processes, the next logical step is to analyse what they are doing. Analysis should identify key elements, such as Job Control Libraries (JCLs), process dependencies, and, most important of all, determine how much 'dead' or redundant code is present in order that this can be removed, reducing maintenance costs.

Having used some or all of the discovery options referred to above to capture source code information and render it understandable and readable, the DMS SRT can then cross-index the enterprise software system, so that source code can be browsed, and each identifier can be hyperlinked to its appropriate target according to language rules. Although discovery activities can be launched using the SRT 'out-of-the-box', analysis requires configuration, and a good understanding of the system under examination.

Although simplified and supported to as great an extent as possible, the analysis stage of any enterprise software project will always remain a time consuming and complex exercise. Given that the goal of Semantic Designs is to capture and subsequently utilise the intelligence that led to the creation of the software in the first place, it is natural that this vital stage is acknowledged as being time consuming. In some cases, several months of analysis may be required to determine what is business logic and what is not, for example. With conventional approaches, this is necessarily accomplished manually. With the Semantic Designs approach, this is accomplished by investing effort in defining the analyses to automate the extraction of this information, so that it may be applied repeatedly and reliably, rather than manual investment. It is a refreshing change to see that Semantic Designs acknowledges that software projects in this class will be lengthy and difficult, and in spite of the quality of the tools it has developed to date, it makes no pretence that the task can be made easy.

Supporting features include:

- Multi-pass attribute-evaluator generation from grammar, allowing arbitrary analysis to be specified in terms of concrete grammar.
- Language-dependent name lookup and namespace management rules are supported by the provision of symbol-table construction facilities, which may be global, local, inherited, or overloaded.
- The next release of the SRT will feature a control-flow graph construction and data flow analysis framework.

The DMS SRT is designed to accept definitions of arbitrary computer languages, along with definitions of arbitrary analyses and of changes to the source code itself, which may be conditional on the results of the analysis stages. One result of this is the ability to identify specific attributes of code, for example, code that is no longer serving its original purpose (or indeed any purpose) and then to apply change across the entire enterprise against all instances of that code. In this example, the result would be the identification and subsequent removal of dead code. The CloneDR option available for use with the SRT is a duplicate-code ("clone") detection facility, which usually finds between 10-20% redundant code in large systems which can be removed.

▶ PRESENTATION LOGIC

The problem with legacy code is typically not that it is substandard, or inadequate for enterprise needs, but rather that its capabilities must be accessed via difficult-to-use interfaces. Application Modernisation tools may address this issue by dedicating functionality to the presentational layers of legacy solutions, in either an invasive or non-invasive manner.

This can be a complex point to address. The majority of solutions focus on delivery of an interface for the benefit of a business user, where user-friendliness and reduction of required skill levels becomes a priority, and in such circumstances support and ease of use are key.

As a tool for use by experienced software engineers, the use of presentation logic by the SRT requires a different focus, which must also take into account the variety of entities (employees, network devices, and applications alike) that are being interacted with by the software being examined. Presentation formats will vary considerably dependent upon the nature of the accessing entity, and the SRT is nevertheless capable of tracing manipulation commands from the source code in order to construct a virtual view of presentation formats.

The next release of the SRT will feature control/dataflow analysis to enhance the product's ability to manage these points, but at present, pattern recognition is used to look for idiomatic constructions. Domain language and abstractions are used to represent the presentation being shown to the application end-user, for the benefit of the software engineer using the SRT.

▶ BUSINESS LOGIC

Separating the presentational layers from the business logic layer can be an equally important element. Business logic, when clearly defined, can be componentised for extensive re-use across processes, simplifying the development of new services and enterprise initiatives. The nature of legacy code, often obscure to contemporary developers, demands strong support in the enablement of business logic extraction and conversion.

The scale of the enterprise legacy code problem can be vast. At least one Semantic Designs customer was obliged to manage over 100 million lines of COBOL, an appalling task if carried out using entirely manual approaches. Semantic Designs approaches this complex and demanding task from the point of view of what it terms 'Design Maintenance'. The aim is to capture the design knowledge that went into the creation of an enterprise system, typically a vast, confusing, and complex artefact, and then to automate that knowledge in order to simplify the enforcement of systemic changes, whether large or small in nature.

One of the considerable strengths of the Semantic Designs approach is the ability to clearly identify each element within a previously opaque legacy application, and to support the generation of reliable impact analysis upon proposed (or accidental) changes to business logic and processes alike. The SRT enables business logic to be abstracted for more flexible usage, as defined below.

▶ BUSINESS LOGIC USAGE

The aim of capturing and improving access to the underlying business logic is to ensure that the business rules that depend on this can be more flexibly supported and utilised. Business rules must also be identified and documented in order to simplify any subsequent recombination into new services. One logical option is for this information to be stored within a business rule repository. The Application Modernisation tool should also be assessed in terms of how it achieves conversions from legacy code, such as which languages it operates with, and the question of how this conversion is carried out in real time should be addressed.

Once identified in the design capture stages, abstracted business logic can be extracted as code fragments, or as parameterised code, and can be located for inspection. Cloned code detected by CloneDR is often more easily understood by software engineers, making it more useful business logic. Detected clones can then be extracted as a parameterised procedure.

The means by which business processes can be maintained during the project will depend on the scope of the reengineering being considered. Analysis and modification can be carried out off-line, and then tested against a snapshot of the system source, extracted from the customer configuration management system. Multiple trials are likely to be needed, owing to the inevitable complexity of the target system, and the fact that these can be accurately repeated off-line minimises the risk of enterprise disruption.

▶ COMPONENT MANAGEMENT

Breaking legacy processes into reusable and highly flexible components is the goal of most Application Modernisation vendors, as this step alone greatly improves application development efforts, and also reduces operational costs. The level of componentisation varies considerably from vendor to vendor, according to focus and core strengths.

Component management is a key area to vendors and end-users positioning themselves to take advantage of the Web services model. As a vendor enabling a range of code-related projects, Semantic Designs does not focus specifically upon Web service implementations, but the clarity of design understanding that could result from using its tools would contribute well to the creation of such a framework. The Semantic Designs approach aids in the capture and identification of components; how these are subsequently managed will depend on the nature of the software project being undertaken by the customer using the DMS.

▶ DEPLOYMENT

The SRT will run upon a Windows workstation with a 1Ghz processor, and 256Mb of RAM, and requires an access path to the software configuration database. Although the discovery elements of the product will function out-of-the-box, the intent is to customise the tool to the point where it is closely aligned with enterprise software, and becomes able to identify and solve problems. It would therefore be inaccurate to define it as an out-of the-box solution, and Semantic Designs makes no efforts to do so.

The DMS itself is designed to work on very large-scale source systems, typically those that possess several million lines of source codes or specifications, across tens of thousands of files, and written in multiple languages. DMS is hosted on Windows NT/2000/XP, using Intel or AMD single or Symmetric MultiProcessing (SMP) hardware.

The expected user of the SRT is a well-educated software engineer, probably with a compiler technology background, but if such expertise is not available then Semantic Designs can provide training in the use of its products; under the right circumstances, there is also a possibility of a service-based undertaking.

The DMS is implemented using the Semantic Designs parallel language, PARLANSE, which provides computational power at an appropriate scale. Large scale software systems require surprisingly large amounts of CPU to analyse and change. Whilst DMS runs upon a single processor system at unit speeds, it can also operate upon symmetric multiple processor workstations with enhanced performance levels. Semantic Designs cites the example that the attribute evaluation process is automatically run in parallel in such circumstances, and can provide a linear speedup on an N-way SMP system.

The available list of languages and complex legacy grammars that the DMS can operate with is very extensive indeed, and space only permits a partial listing here: ANSI C++, C#, COBOL 85, and IBM VS COBOL II with CICS/SQL, Fortran 95/90/77, HTML 4.0, XHTML, plus Internet Explorer dialects, Java 1.1, 1.2, 1.3, and 1.4 (This includes full name and type resolution), IBM JCL, Jovial 73, PARLANSE, Pascal (ISO 7185) and Borland Delphi, PL/1, Progress, SQL (ANSI SQL2/SQL 1992, Oracle8 SQL), Rational Rose Universal Modeling Language (UML), Visual Basic 6, VBScript + ASP, and XML.

Additional language modules and grammars are available from Semantic Designs, which states that it is willing to define other languages that are appropriate to its market.

MARKET STRATEGY

Semantic Designs positions itself as a product vendor, with the primary goal of training its customers to make use of the DMS approach for themselves, it also considers proposals for service opportunities related to the DMS. In Butler Group's opinion, this is a sensible strategy, given the complexity of any enterprise software project; the vendor's expertise will in almost all cases exceed that of the customer, and such familiarity will prove to be a compelling element of a service usage scenario.

Issues with application code, particularly that residing upon mainframe systems, affect both horizontal and vertical businesses. Semantic Designs identifies four key targets for its products:

1. Large IT shops, with enough internal resource to keep reusing the DMS on numerous application projects.
2. Customers faced by the need to make an enterprise migration, either with or without support from service partners.
3. Service companies, which may provide reengineering services.
4. The low-end software engineering tools market.

The HP3000 migration space is an obvious opportunity for Semantic Designs, along with other companies that provide migrational support; the phasing out of this system is a major event in the Application Modernisation market. Less extreme examples of opportunity lie with companies attempting to migrate some, or all, of their legacy applications onto more accessible platforms. In addition, it should not be forgotten that difficulties with aging code can also be addressed by transformation, and such specific issues can readily be resolved using the DMS SRT.

Semantic Designs currently uses direct sales channels to reach its markets, and expects to develop service partner channels in due course. There are no requirements for business or technology partnerships to support its products at this time. Each development seat costs US$25,000, while the predefined language modules vary between US$5,000-20,000 depending upon the complexity, maturity, and demand for the type(s) involved. Internal runtime deployment of custom tools runs to US$2,500 per seat, while in service contracts there will be a charge of between one and four dollars per line of converted code. Annual maintenance is 15% of the license, and this includes upgrades and technical support.

KEY FINDINGS AND LOOK AHEAD

KEY FINDINGS

- ✓ Exhaustive list of language modules and grammars.
- ✓ Automated reading of several million line software systems.
- ✓ Interesting focus upon code design, instead of maintenance.
- *i* Targeting public and private sector mainframe and code issues.

Key: ✓ Product Strength ✗ Product Weakness *i* Point of Information

LOOK AHEAD

Semantic Designs is pursuing a strategy of aiding very large scale enterprise migrations, in the public and private sectors, in order to raise the visibility of the company and its products. Even more languages are being added to the SRT, and other products are in development to manage issues such as metrics, profiling, automation of bug detection, and code generation.

COMPANY PROFILE

Semantic Designs has its headquarters in Austin, Texas, where the company was founded in 1996. The company was founded by Dr. Ira D. Baxter, the company's CEO, using an initial grant of US$ 2 million from the US Department of Commerce NIST Advanced Technology Program. Four years of research and development led to early revenues being generated by consulting, custom software projects, and working with early adopters. Semantic Designs has concentrated on the productisation of the DMS, and has undertaken concentrated commercial efforts since 2002.

The company is privately held, and reports revenues of between US$600,000 and US$ 1 million dollars per year. At present there are four employees operating out of the office in Austin, although Semantic Designs expects to take on another 8 to 20 personnel this year, mostly to handle service contracts. Enterprise customers making use of Semantic Designs technology include Northrop Grumman, Rockwell, Salion, and Tivoli.

▶ CONTACT DETAILS

Semantic Designs, Inc.
12636 Research Blvd.
C214 Austin
TX 78759-2239
US

Tel: + 1 800 367 0482
Fax: + 1 512 250 1191

E-mail: info@semdesigns.com

www.semanticdesigns.com