Written by Michael Thompson, March 2002 TECH/RPS/1053

Technology Infrastructure

Research and Advisory Services

Application Development **RESEARCH** PAPER

Lazy Software Sentences v2

Abstract *Sentences is an ADE based upon a radically different database model that allows development of applications and the underlying data model to take place simultaneously. Sentences answers the need to develop applications quickly and simply without using dedicated skilled resource. The underlying data model is powerful in a business sense as it reflects the real world in its treatment of entities and associations between those entities. The underlying data infrastructure of all business are built on the relational model and nearly all the technologies implemented refer to that model. The benefit of considering Sentences would be felt by any organisation that wants to develop and deploy personalised applications. Lazy Software has introduced a reduced pricing structure allowing a three month trial of Sentences with the right to keep and use any developed applications without further cost. The speed of development inherent with Sentences makes this a strong economic proposition.*

STRENGTHS

- Answers the restrictions based on development by the constraints of the relational model.
- Speed of development.
- Relevance and personalisation of the developed applications.
- Interfaces with XML.

WEAKNESSES

- Pervasiveness of the relational model inhibits uptake of new paradigm.

FUTURE POTENTIAL

The future of Sentences may well lie in tighter tie-in to XML to give access to further resources, and the creation of custom applications built on the Sentences engine.

▶ FUNCTIONALITY

Product Analysis

Sentences is a data-focused application development tool that works using the Associative Model of Data (AMD), which allows the underlying database to be 'built' as the application is developed. The AMD is radically different from the relational model that is pervasive throughout organisations in that it takes a real-world view in terms of the data being utilised, without the restrictions that are inherent with the relational model.

Typically, applications are developed to use data from some underlying pre-implemented data source and as such are restricted by that data model. Sentences is radically different in that the developer, and in this instance this should not be seen as a specific individual with typical development skills, creates an application that answers a specific need using a simple drag-and-drop interface.

To better understand how this development takes place, it is necessary to understand some of the terms used within Sentences and how the AMD works. A Sentences database contains two elements – Entities and Associations. An Entity is an object that has an independent existence and does not rely on any other object for that existence. An Association, on the other hand, is an object whose existence is reliant upon one or more Entities or other Associations.

From this simple model it is possible to build truly dynamic databases that can grow and adapt as the need arises. One way of considering this is to see the AMD as a data model that allows changing context to be loosely coupled to data elements. As an example, a common use of the relational model is in the creation of a customer database that applies constraints from the moment of its design as it takes an entity and contextualises that entity by pre-defining its relationship to another entity – the business. Therefore, in the relational world, an entity exists within a single role, and if the same entity has a different role then a different data structure has to be built to reflect that role.

Sentences, would take an individual as an Entity, could create another Entity – such as a business – and then form an Association between the two that defined the relationship (in the non-database sense) between the two Entities. This association could be defined as a customer or it could be defined as a supplier, based upon the specifics of the defined relationship. The two entities remain the same, but the interaction between them can be redefined without the need to change the underlying data elements that describe the entities.

Whilst this might sound complex, the complexities are hidden from the end-users, whose only task is to define the entities and the associations using the GUI. The use of the GUI is also relevant to the retrieval of information through searching, and Sentences has extended search capabilities that are not restricted to an SQL-type input.

From the development point of view, it is worth looking at what the benefits really are. There is an inherent imbalance between the requirements of application development and the structure of the underlying databases that the vast majority of the developed business application programs need to access.

This imbalance is caused by the fact that application developers – or programmers – are most effective when they can work with simple database schemas, while the major concern of organisations is to have a correctly constructed database implementation, in terms of the creation of normalised databases, this means that they are necessarily complex. This complexity creates a problem when it comes to the development of applications, and the writing of application programs.

Application developers are not necessarily database experts – in fact in many cases they will have little understanding of how the databases operate – but they need to be able to manipulate the data in a manner that ensures the applications perform the correct functions, and that they do not adversely impact on the database in terms of performance or data integrity.

One favoured solution is to remove the business logic layer from the database layer, which allows application programs to be changed without impacting the database. While this solution is adequate and addresses many of the problems, it is a solution that has been forced upon businesses due to the relational database model. The downside of this method of speeding up application development is that it adds another layer of complexity, and an additional layer onto architectures that are already becoming over-complex.

The other alternative as presented by Sentences is to get around this conflict by getting rid of the root cause of the problem – the structured database. Once the problem is looked at from this angle, and an ADE provided, then there are no restrictions on who can develop applications, nor are there restrictions on the size and use of those applications, other than the granting of access and modification rights.

Product Operation

Contents of Sentences databases are stored in Chapter files, and these Chapter files can be configured to make up a Profile, which is a specific database. Therefore it is possible for individual users and applications to have access to different Chapters that would create a new Profile. Hence the removal of the requirement to rewrite applications for each required instance of data access and usage. The issue of clustering is taken care of by allowing separate chapters on different servers and a whole Profile can be assembled by using a URL to access the various required Chapters.

There are three main elements to the Sentences GUI:

1. The Explorer.
2. The Dataform.
3. The Query Editor.

The Explorer

The Sentences Explorer is the principal workspace, and has the same look and feel as Windows Explorer, with the same range of extensible features; such as how the panes are displayed, toolbars, menu bars, etc.

It is within the Explorer that database schemas are created by dragging and dropping entity types and associations. As there is no referencing to underlying tables, there are no restrictions imposed by keys. Thus, for an example, a person entity can be a customer (by association) or a supplier (by a different association). For anybody used to having to build database schemas and then write applications (or programs) on top of those schemas, it is almost impossible to describe the ease of use that Sentences provides.

While it takes a different set of skills, those skills are not based on the ability to learn some syntactical language, but rather on simply understanding what is required from a business perspective, what entities will be involved and the associations that need to be created.

The Dataform

The Dataform is used for updating and displaying the associations that belong to an entity or association. The Dataform can also be used to create new entities based on an existing entity type – extending the principle of inheritance in pure application development terms to the database. The Dataform also provides a presentation environment that is built 'on-the-fly', so that data elements (to use a relational analogy) can be presented in any form required. There is also a stronger tie-in between the Dataform and the Query Editor in v2 than in v1, giving a much more structured user-interface

The Query Editor

The Query Editor is used to create and access queries. Sentences has its own query language, which in v1 was very much SQL-based. This was considered to be a weakness by Butler Group, and it has now been extensively revised to provide the sort of functionality that is available in the rest of the product. SQL-based set querying can still be carried out, but there is now a more intuitive drag-and-drop interface that creates meaningful queries.

Created queries can build dynamic Dataforms for the presentation of the results and the query can be exported to XML to supply the Document Type Definition (DTD) for style sheets that can then be reused as the default for returning queries.

Overall, the major enhancements to v1 concern the tie-in with XML, with support for bulk import/export via XML as well as CSV files, and importation can be done from HTML through DTD.

▶ DEPLOYMENT

The supported platforms for Sentences are:

- Microsoft Windows NT4.
- Microsoft Windows 2000.
- Microsoft Windows XP.
- Linux.
- Sun Solaris.
- IBM AIX.
- IBM OS/400.

Supported Web Servers are:

- Microsoft IIS 4.
- Apache.
- Tomcat (supplied with the product).
- iPlanet.
- IBM WebSphere v4.

The Sentences Server runs as a Java servlet installed as part of a Web server, and runs automatically to process requests received from the Sentences Client that runs as a Java applet in a Web browser. The only installation required on the client is for the Java 2 Runtime Environment (J2RE) plug-in for the client Web browser.

Installation is carried out from an HTML page that can be modified to fit into operating requirements. The changes that can be made include:

- Changing the port to which the applet connects to the Sentences Servlet. By default the connection is to the same port from which the HTML page was downloaded. If the server is accessible via a different Web server then the port needs to be specified.
- Although Sentences uses and recognises standard Java datatypes; such as Number, Timestamp, and Text; if a Profile uses any custom datatypes then the class names have to be specified.
- The default number of items sent for each request can be amended.

Installing Sentences is as simple as that, and the only further requirement is to train the users, who will effectively be carrying out the development work. Lazy Software provides a three-day foundation course and there is on-line help available. When that is compared to the time taken simply to 'hook' an ADE into an underlying database, some of the benefits of Sentences become instantly obvious.

Custom datatypes can be created by anyone who is proficient in Java by writing Java classes that implement the Sentences Datatype interface. Both custom and supplied datatypes control the behaviour of an entity type. Datatypes cannot be changed by an end user and custom datatypes appear to the user alongside the supplied Java datatypes.

▶ PRODUCT STRATEGY

For Sentences to become successful, Lazy Software has many problems to overcome. The first is that as a pure ADE it is up against well-established products that have a large installed base. As a new data model it would have to take on the corporate reliance on relational technology. Therefore, central to the success of Sentences will be tying these two elements together to demonstrate that ADEs and relational databases are not the only way that business problems can be solved.

It is attempting to get this message across by having a direct channel in the UK and forming distribution agreements in the rest of EMEA. Another possibility exists that application developers can build custom applications on the Sentences 'engine' and market them through a royalty-type agreement. This is a model that has been successful in other areas in the past.

The benefits of Sentences are relevant to businesses of all sizes, but currently Lazy Software is targeting the enterprise market to try and gain acceptance for what could be a revolutionary product.

▶ COMPANY PROFILE

Lazy Software is an independent, privately-owned company formed by Simon Williams who was founder, Chairman, and Chief Executive of Synon from 1984 until 1990. After 1990, and the restructuring of Synon, which reflected the success that the company had had in the ADE market, Simon became CTO. After the acquisition of Synon by Sterling Software, Simon left to develop the Associative Model of data and build Sentences D-FAD.

Currently with offices in the UK, and some 42 personnel, Lazy Software plans to promote Sentences both in EMEA and in the USA where it has set up offices in Chicago and Dallas.

▶ SUMMARY

Butler Group believes that Sentences from Lazy Software is a solution that the market has been waiting for, even though the market may not have realised the fact. What might have been an exercise in database advancement – an academic exercise – has been brought into the business arena by a clear understanding of the shortcomings of both the relational model and the object model.

This business focus has been extended by the creation of the ADE, which will ensure that users can start to benefit from implementing this solution without a steep learning curve. While Butler Group believes that there will be few organisations that will move away from the relational model to embrace this new paradigm – especially in the initial stages – it also believes that businesses that are open-minded enough to consider radical solutions will be pleasantly surprised at both the ease of use of Sentences, and the business benefits that it will bring.

▶ CONTACT DETAILS

Lazy Software
Mercury Park
Wycombe Lane
Wooburn Green
Buckinghamshire
HP10 0HH
UK

Tel: +44 (0)1628 642300
Fax: +44 (0)1628 642301

E-mail EMEA: info@lazysoft.com
E-mail USA: contact@lazysoft.com

www.lazysoft.com

For more information on Butler Group's Research and Advisory Services contact:

Europa House, 184 Ferensway, Hull, East Yorkshire, HU1 3UT, UK
Tel: +44 (0)1482 586149 Fax: +44 (0)1482 323577 www.butlergroup.com

Published on behalf of Butler Direct Limited by Addax Media Limited www.addaxmedia.com