Review: Cadet

reviewed by Mike Orriss

Cadet (Computer Assisted Database Engineering Tool) is a low cost yet powerful database engineering tool with a graphical user interface for building complex databases. Cadet's interface is very powerful and easy to use, designed for both novice and expert users. Cadet provides a visual representation of databases regardless of the system architecture. Cadet also automates the database engine development.

Well, that's what the introduction states in the Cadet manual, but what does it actually mean? Cadet is definitely a low cost entry into the world of CASE (Computer Assisted Software Engineering) but does it work in practice and is it worth buying and learning? Having already both bought and learnt Cadet, this review describes how I use it in practice so that you can judge for yourselves.

CASE tools come in all shapes and sizes and vary enormously in functionality. There are products covering just the start of the design cycle, while others generate and maintain complete applications. Cadet is definitely of the former variety, although there is limited reverse engineering and application building supported. Cadet is still a very young product, though, and I fully expect more of such functionality as it matures.

Installation

There are three flavours of Cadet available: Desktop with Paradox and dBase support only, Standard which adds Interbase, and the Professional version which includes support for Client/Server back-ends such as Oracle and Sybase. As there is a significant price hike for the Professional version, I opted for the Standard and this review is based on that.

Cadet arrived on four diskettes containing Setup for Win95, Setup for Win31, a tutorial and a manual (in Word format). I installed the Win95 version with no problems and then installed and ran the tutorial (twice).

I have to confess at this point that I did not even unzip the manual and have only done so just now in case there was anything that I had missed which was worth reporting. There wasn't – it is just a recap of information that I found quite easily via the tutorial and help file.

Tutorial

The on-line tutorial does a good job of demonstrating the basic techniques of the product. It's basically a guided tour of the facilities, driven like a video recorder with fast forward and backward allowing you to skip to the start of sections. Once I had run through it a couple of times and then experimented with using Cadet myself, I felt perfectly at home with the product.

Concepts

Cadet basically provides a drawing surface where you design your database model by adding *entities* (think of them as tables) and connecting them via *relations* (or indexes). Entities consist of a set of *attributes* (fields) which can be keyed or non-keyed. One point to note here is that when you are defining a relation to connect two tables, the key field(s) of the primary table are used and the foreign key(s) of the child table are generated automatically.

Re-Engineering

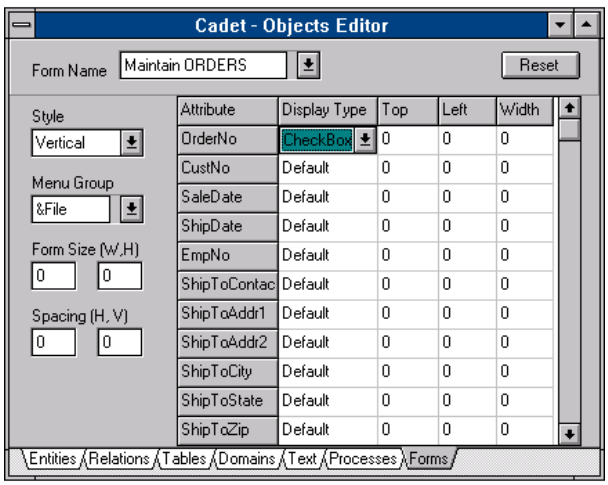
In order to produce screen shots for this review, I used the Build/ReEngineer menu option. This will generate a model for all the tables within a directory or alias.

I have found this option extremely useful for understanding and printing the structures of existing database systems which arrive

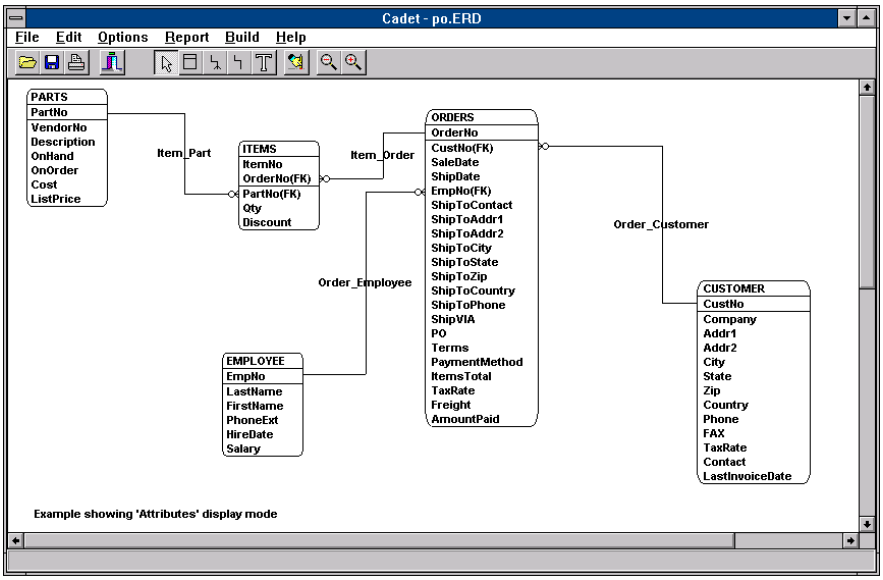
➤ *Table 1: Data types supported by Cadet*

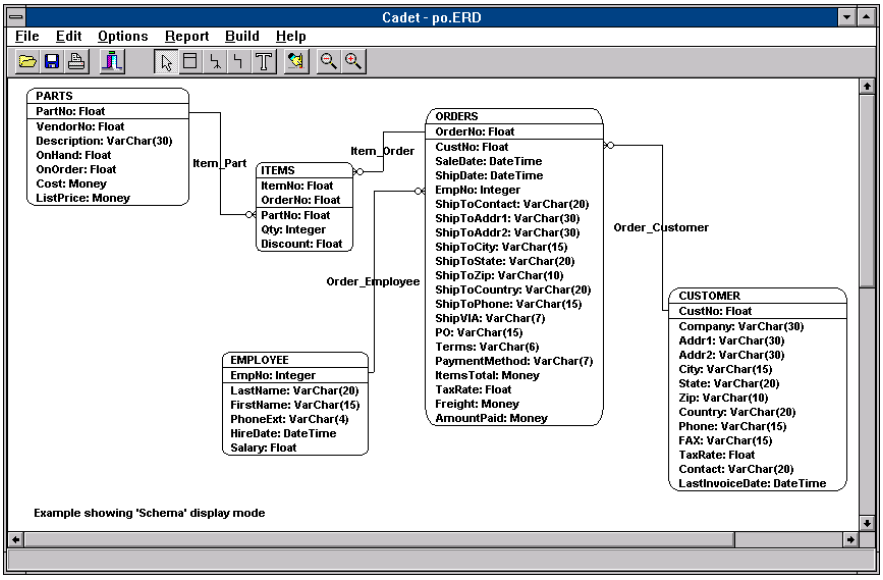
Binary	Text	TimeStamp
DateTime (see note in text)	Boolean	Char
Long Binary	Double	Int
SmallInt	Money	Number
Blob	Time	VarChar
Decimal	Byte	Date
Long VarChar	Float	Integer
	Numeric	Real

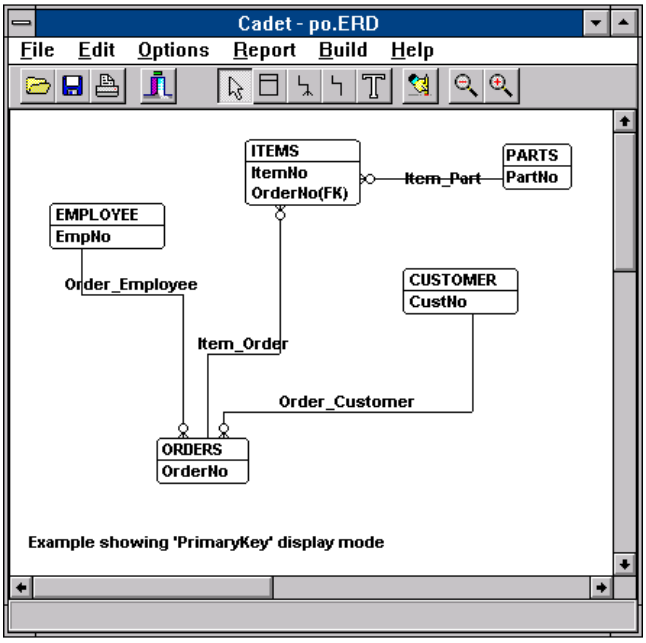
➤ *Cadet's Objects Editor*

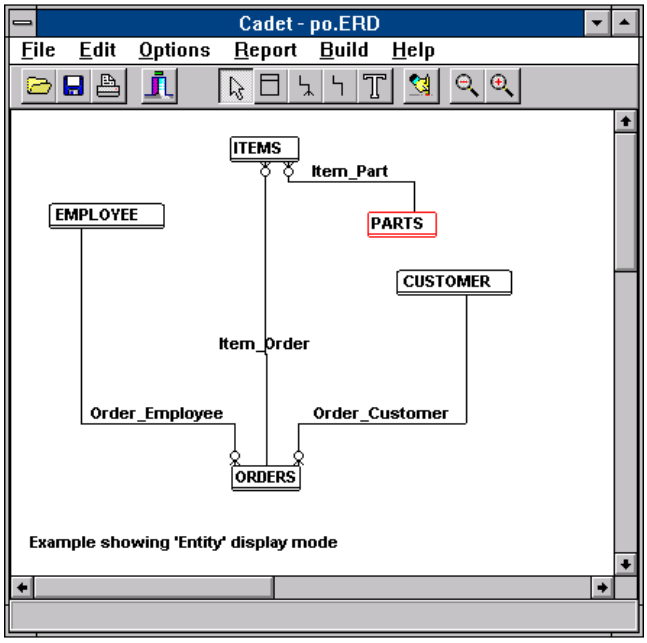


➤ *The four screenshots on this page illustrate the four Display Mode options available in Cadet: Attributes, PrimaryKey, Schema or Entity*









without sufficient documentation (and there are plenty of those!).

All attributes (fields) default to `Char(10)`, so the next step is to change their types to one from the variety available (see Table 1) or by specifying a *domain* name (which I'll explain later).

Once this is done, the set of tables, complete with primary and secondary indexes, can be generated for the directory or alias of your choice. You can select which tables to generate and I have found this to be extremely useful, as this allows you to change the design without having to regenerate all the tables.

It does not, however, generate relations from existing indexes, which is a great pity. The examples shown are the result of loading the DBDEMOS tables provided by Borland, deleting non-required entities and adding relations.

Cadet also offers facilities for generating forms and applications. I have played with this, but have found it to be a bit simplistic and prefer to build my forms myself.

Display And Printing

The amount of information which is displayed is controlled by selecting one of the four Display Mode options: Attributes, PrimaryKey, Schema or Entity, as illustrated by the screen shots on this page. Note that I have adjusted the layout for each one. All the items may be

dragged around the screen as required.

Text may be placed anywhere on the model and the page size and font can be changed. I would like to have the option of a different font for each screen element, however *(are you reading this Samson?)*. Zooming in/out is supported.

A model may extend over multiple logical pages and you can select which page(s) to print. By the way, this product really gains from a high screen resolution: I normally develop at 1024x768 and I really noticed the difference when preparing my demo for the UK *Delphi Developer's Group* at 640x480.

There are two reports available. The Entity Report allows you to specify just how much information you want to print and is a very easy way to document the model. Incidentally, descriptions can be added to all elements of the model, including individual attributes. This makes the model an ideal place to store information. The Consistency Check Report displays on the screen with an option to print. It checks that all attributes and domains are declared and is useful because it is possible to use domains before they are actually declared.

Editing The Model

Delphi developers will have no difficulty here. Creating a new element is done by right-clicking on the model and selecting the element type. To modify an existing element, just double-click on it. Both actions invoke the Objects Editor, with the correct notebook page current.

The model is live, so any change made in the Objects Editor is immediately reflected, although you do have to move off the changed field for it to take effect!

Domains

Domains are the main reason why I use Cadet instead of the DataBase Desktop to generate new tables. A domain is a user defined field type. It allows you to specify global types, thus allowing a single change to affect many fields. Another feature is the ability to declare a validation rule. Building a model using domains offers significantly easier maintenance and a way of designing and documenting a database that is reasonably independent of the back end.

Wrinkles And Gotchas

This is, as I said before, a young product and there are a few points that could do with a bit more work. The main inconvenience I have found relates to TimeDate types: they actually generate Paradox Date fields. This problem has been acknowledged and I am expecting a fix shortly.

Some names for fields are not accepted. For example, you cannot declare Date as the name of an attribute, which some may regard as good practice anyway, but it can cause problems when building a model from an existing set of tables.

Another minor inconvenience concerns domains. The drop down list of types only shows the predefined types and you have to key domain names in manually. I tend to use domains for all field types as this offers simplified maintenance, so this lack hits me hard.

The auto-generation of foreign keys is nice, but an option to link tables by existing field(s) would simplify model building using existing tables. Currently, you have to delete the existing attribute from the child table.

Summary

Although Cadet has a few rough edges, there is already sufficient functionality to make it a good purchase for many. I am not so convinced that it is the right tool for Client/Server, as the extra cost brings it into competition with established rivals like S-Designer. I am happy with it for the time being and I am reasonably confident that the enhancements due this summer will keep me on board.

Cadet is available from:

Objectware
2951 N. La Chiquita Ave.
Tucson, AZ 85715, USA
Email 102522.1523@compuserve.com
http://ourworld.compuserve.com/homepages/objectware

Prices are: Desktop $95, Standard $185, Professional $495 (including shipping and handling). The Standard version is available in the UK for £140 plus VAT from the Delphi Developer's Group (Tel: 01980 630032).