The ERROS Connectionist Database

An Innovative, Productive, Simple to Use, Alternative to The Inflexible Programming Paradigm

Abstract

The innovative, patented, NoSQL ERROS Connectionist Database provides an incremental way of creating powerful applications for business and the humaniities, without detailed upfront user or system specifications, without physical file design and mostly without any program coding or program generation. Database and application structures are defined in the user's natural language rather than in program code. Security can be at the field level. No new files are created during application creation and there are no compiled ERROS applications.

The ERROS database handler interprets the definitions and accesses and updates all metadata and all user data for all ERROS applications. Records can be of infinitely variable length, without null values, and attributes are repeating by default. The database can combine very complex multidimensional, relational and network data structures with limitless nested levels of hierarchy, without redundancy. Application development is largely self-documenting and changes can be rolled back.

The almost zero second response times do not noticeably change as file sizes grow.

The ERROS Connectionist Database is automatically indexed with a unique multipart key structure. There can be limitless bidirectional connections between any records of any type. ERROS turns raw data into a fully navigable semantic network. Users can browse the bidirectional connections in either direction, with the same immediate response, without using joins and without using a query language. Data can also be stored about any connection or relationship as can lower level nested connections.

ERROS generates HTML and Javascript on the fly and can be used for creating major, web enabled applications, with automatic concurrency control, and including transaction processing, for a wide variety of businesses, and for the humanities.

The ERROS Connectionist Database

Since the beginning of the computer industry, business applications have been built using computer programs. These contain the business rules and a database stores the data to which these rules apply. These separate worlds of data and programs have to be kept in synchronisation, at best a fragile process as it requires two different disciplines – program coding and database design. This separatism – the splitting of the data from the rules – is responsible for most of the unnecessary complexity, high costs and extended time scales of application development and maintenance.

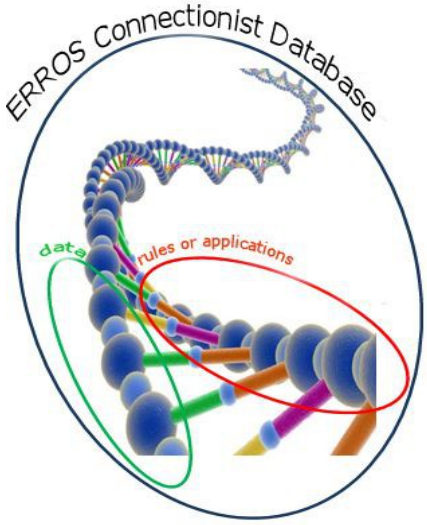
Making even simple changes to a traditional, separatist system can be a problem. Adding an extra field to an existing table may seem straight forward, yet implementing such a change can be quite challenging. This is because of the architecture of traditional applications, which, with their rigid database systems and computer programs, are not easy to change.

Application creation has always depended on computer programs, created by programmers or generated by a computer. Program coding is a very slow process and as program languages are not understood by users, they cannot be sure that a system will meet their needs until it is ready for final testing. They might request changes that they consider minor but which are not easy to implement.

Much energy has been expended in trying to speed up the program creation process, but it seems that few people have stood back and wondered whether programming is the only way. The most efficient data storage and retrieval system that we know is the human brain. This can absorb new concepts with ease and, unlike computer systems, it is not shut down, redesigned, reprogrammed, tested and restarted to accommodate even the simplest change. If this were the way that our brains were updated, we would know almost nothing. It seems that the brain's astonishing capabilities are in part dependent on its accumulated knowledge, and its ability to navigate the connections between individual pieces of data. Our brain is clearly data driven and it probably does not make much distinction between the rules, which we might call metadata, and the data to which those rules apply – in computer terms, the user data.

Why not follow the brain and replace program code with records in a database? Database updating is dramatically simpler than program coding. With the ERROS Connectionist Database, all metadata, user data and application definitions are stored in a unique, open-ended, totally flexible, NoSQL database management system that is based on Artificial Intelligence or cognitive techniques. Its advanced automatic indexing makes it into a fully navigable semantic network in which all data is stored in its context. A relatively small program kernel, the ERROS database handler, interprets the definitions and accesses and updates all metadata and all user data for all ERROS applications which include the ERROS Business Modeller, the ERROS Application Creator and the ERROS Authority Organiser. These are used to build all other applications, including those created by users.

In all ERROS applications, the data and application definitions are created employing object-oriented techniques that allow reuse of all data definitions. These are stored together with menus, procedures, application and security definitions and the user data in the ERROS Connectionist Database. Screen and printer layouts are also stored in the database and multiple views of all data can easily be created. By navigating connections, the views can combine data from multiple entity types, even on the same line or row if required.



The ERROS Connectionist Database might be compared to the structure of the DNA string in which both sides are related but independent and are interconnected to form one whole. In ERROS, the data and application definitions and the user data are integrated and always synchronised.

ERROS entity type, attribute and application definitions are stored as connections, defined in the user's natural language (English, French, German etc.), rather than in a programming language that only computer specialists can understand. When new entity types, attributes or applications are defined, no new files are created and no program changes are normally required. Metadata and user data, with little distinction between them, are stored together in the ERROS Connectionist Database in a Universal Data Type and are updated and retrieved using the ERROS database handler. An ERROS application developer simply gives a new entity type or attribute or application a name and adds it to the database. The ERROS Connectionist Database is defined in itself, using the same structure as for all other definitions.

Most attribute values, whether user data, meta data or images, are stored as bidirectional connections and users can access any data to which they are authorised by navigating these in either direction at the same very high speed, allowing them to find connections that they did not know existed. No joins and no query language are required. For ERROS turns raw data into a semantic knowledge base. Like the brain, ERROS applications are data driven. Attributes can store unlimited text.

The ERROS Connectionist Database can combine very complex multidimensional, relational and network data structures with limitless nested levels of hierarchy. "Children" can have multiple "parents" and users can travel up or down hierarchies and enter and exit at any level, not simply at the top. No joins are required. ERROS can handle totally variable length data without Null values or wasted space. Menus are stored in the Connectionist Database and can also be nested.

ERROS allows genuinely incremental development of complex systems, without a detailed upfront user or system specification and without detailed physical file design. Application development is largely self-documenting and changes can be rolled back. Changes to the database structure and the applications can be made whilst the system is live, without impacting the performance of other users on line at the time. All ERROS applications are developed and can be operated over the Internet using the ERROS Standard Operator Interface (SOI) which dynamically generates HTML and JavaScript to provide a Graphical User Interface for PCs, Macs, tablets or smartphones. Developers do not require HTML skills. The GUI adapts immediately to accommodate any changes. SOI also generates PCL5 for printing.

ERROS is a totally new way of creating major computer applications. Application development and maintenance are dramatically simpler than with traditional methods and have a short learning curve. ERROS is extraordinarily productive. ERROS applications, that can be changed on demand, are developed –

- incrementally - you can 'grow-as-you-go',
- without a detailed user or system specification,
- without physical database design (i.e. no normalisation),
- without any new file creation,
- mostly without program creation or change,
- with considerably reduced development, maintenance and operational costs,
- and operated using a browser over the Internet.

There are no compiled ERROS applications and the concept of compiled, rigid, inflexible applications has gone.

ERROS solves with ease a variety of problems for which many developers and even database researchers do not have a generic solution, such as

- storing multiple record types in the same table,
- incremental development,
- bidirectional, many-to-many, relationships or connections,
- variable length records without Null values,
- repeating attribute iterations (by default),
- retaining history of updating,
- automatic indexing of attribute values within each entity – for instance employees of a company will automatically be displayed and can be retrieved in the sequence surname/first name(s), although displayed first name/surname, and also in employee number sequence. Sales or purchase order lines can be displayed in order of product name or product number. No coding or sorting is required.
- unlimited nested levels of hierarchy, allowing groups or sets,
- storing connections with connections,
- allowing any entity to belong to multiple entity types and inherit the attributes of the entity type under which it is being considered.

ERROS allows entity types, individual entities, attribute definitions and individual attribute values or iterations to be referenced, stored and retrieved by name, and/or number and/or date (and time), without any regard to their physical location. Any attribute may contain one or multiple fields. The attributes in any record may contain both 'public' and 'private' (i.e. user) data for collaborative applications. Users retain secure control of their data. An ERROS application developer does not need to design a physical database schema and indeed cannot do so. ERROS also allows multiple alternative record identifiers or synonyms that can be used to retrieve records, perhaps where their name has changed.

ERROS can be used for creating major, web enabled applications, with automatic concurrency control, and including transaction processing, for a very wide variety of businesses, and for the humanities, for cataloguing, collections management, archiving, recording history of any type, and for ontology and taxonomy. ERROS is very suitable for creating application packages, such as ERP, which can easily be modified to suit each user without changing the base package.

Navigation of bidirectional connections in either direction at the same speed is a most powerful feature that has a dramatic impact on performance, and thus on scalability, and also ease of use, compared with query languages. For instance, if an entity type called 'Telephone Number' stores all telephone numbers known to a business and these are related, with bidirectional connections, to the person or company that subscribes to each number, then, as all records are indexed, to find the owner of a telephone number only requires access of the record for that telephone number, without searching as it is indexed, and the ERROS application will immediately display the name of the contact(s) using the number and information about them. Even if there are, say, a million contacts, each with one or more telephone numbers, this only takes a small fraction of a second. ERROS allows many-to-many connections by default.

If a user is viewing a sales order for a customer and clicks on an order line for a particular product, the ERROS application will immediately navigate to the record for that product and display orders for that product. These would normally be displayed in date sequence with most recent first, but the operator can access the orders in customer name sequence. This is achieved by indexing rather than sorting. Clicking on an order will immediately display details of that order. If, say, the product is purchased from an outside supplier, the user could click on their name to find details of the purchase order and the name and telephone number of the supplier's employee handling the order. No query language is required. Clicking on the telephone image next to telephone number would cause a VOIP application to call the number.

The bidirectional connections can be temporal – i.e date and time dependent.

Most systems use a relational database for the storage of their data and many use the process of data normalisation to model its structure, probably in '3rd normal form' to reduce data redundancy. To build a normalised model of all the data in a business is a slow, tedious process that requires significant specialist skills. With the ERROS Connectionist Database, the process is much simpler and faster, requiring rather less skills – mainly an understanding of the business rather than an understanding of technology. With ERROS, there should not be any data redundancy. It could be said that data is stored in 6th normal form in the ERROS Connectionist Database, but, fortunately, ERROS developers do not need to know what that means, nor do they need to know that surrogate keys are used throughout the ERROS Connectionist Database. They don't need to understand the key structure of the database as ERROS ensures totally consistency in the modelling process.

ERROS has been used to create STIPPLE, a major cataloguing and research system for the fine and applied arts and the humanities. This most advanced collaborative system can be used by any number of institutions throughout the world to create catalogues raisonné and union catalogues and for collections management of any type of object. Each object type has its own data schema. STIPPLE has over two hundred separate entity types, each defined with multiple complex relationships. For objects of which there are multiple examples, such as prints,

sculpture, ceramics, coins and medals, postage stamps, and motor cars, standard information such as name or title, description, engraver, bibliography, states, etc. only needs to entered once, however many examples are catalogued. Institutions put in the accession numbers of their objects, together with other data that only applies to their particular object. STIPPLE can handle enormous volumes of data and images. It has the potential to catalogue all the fine and applied art of the Western world in one integrated system. STIPPLE can produce typeset printed catalogues, labels for exhibitions, etc., or output XML for transfer to a publishing package. All institutions can share biographical records, records in the gazetteer, bibliographic references etc., with much reduced research and data entry costs. STIPPLE integrates the data from all disciplines.

All ERROS applications -

- have exceptional, almost zero second, response times,
- are scalable - performance does not noticeably deteriorate as file sizes grow,
- are suitable for enormous quantities of data,
- are very robust,
- are always internet ready – no content management system is required,
- have outstanding security which can be at the field level,
- have an audit trail that applies to all meta data and application definitions and to user data. All changes to meta data and user data can be rolled back. The audit trail applies to each separate attribute iteration.
- can be changed in line with the ever evolving world of the users,
- can include images
- have a high availability option,
- are automatically integrated, sharing data without redundancy,
- have high data integrity,
- have much lower Total Cost of Ownership for development, maintenance and operations,
- can display or import data from other sources or systems (programming in an 'exit' program that does not affect the ERROS main programs may be required).

Computer hardware has changed beyond recognition in the last fifty years whilst underlying system development methods have not changed, although Graphical User Interfaces may make them appear different. Despite a variety of initiatives, such as Agile, system development still requires detailed file design and program creation using iterative processes. Unlike ERROS, these do not allow incremental development.

In ERROS, the rigid separatism of traditional development methods has been replaced by the flexible connectionism of the real world. The concepts of ERROS have been patented.

ERROS is a total paradigm shift in modern application development methods so cannot easily be compared with traditional development methods nor can it be described in a few sentences. It is not like any other product and any assumptions about how it works might be misleading. ERROS runs under the IBM i operating system on the IBM Power Systems platform and on its predecessors. For an insight into how ERROS works, visit
http://www.erros.co.uk/ERROS_Description_2014-08-16.pdf

Rob Dixon 23rd May 2019
rob.dixon@erros.co.uk