

## Case Study – Corporate Asset Information Architecture

**Topic:** Using kinodb for the development of a corporate information architecture

**Summary:** kinodb was used to:

- Define and implement a corporate data architecture allowing the integration of numerous incompatible legacy asset information systems.
- Support change control processes to enable the coherent management of the data dictionaries of the legacy systems.
- Implement a generic 'universal' asset information repository that could be used to warehouse data from other systems, enabling the retirement of redundant data stores.
- Publish web-distributable details of corporate asset type definitions including data collection requirements, standards and regulatory constraints.

**Business background:**

- A regulated infrastructure maintenance organisation with a multitude of disparate data sources developed over many years.
- Numerous 'home-grown' and tactical asset information sources developed to differing standards.
- Multiple, inconsistent definitions of key corporate asset types with resultant inability to gain a single view of asset information.
- No central management of the data specifications of the various asset information repositories.
- Poor dissemination of definitions, standards etc.

**Challenge:**

- To design and implement a single repository capable of recording the master definitions of asset types, each mapped to the disparate definitions supported by the various existing asset information systems.
- To enable web-based reporting of asset type definitions and associated data collection standards.

- To use these master asset type definitions to drive the implementation of a 'universal' asset register capable of maintaining data from existing and future systems.
- To enable the rules-based cleansing of data in existing systems.
- To enable migration of data from existing systems to enable their retirement.
- All data maintenance to be performed by business users, not IT staff.

#### Solution:

kinodb was used in the design and definition of an architecture for the maintenance of the organisation's asset information. This architecture was based on the central maintenance of the data dictionaries of each of the organisation's core asset information systems, allowing the asset information recorded in each to be related to and from a standardised set of asset type specifications.

Using kinodb's rapid development capabilities, the initial solution was prototyped in user workshops to demonstrate the validity of the proposed data model. This initial implementation was then used as the basis for the development of the production system.

Additional workshops were conducted with business users to determine their requirements for support of change control processes, reporting requirements and so on. Functionality in support of these requirements was developed incrementally with the close involvement of the users, who were able to use early versions of the system for data entry use before the delivery of the completed system. kinodb's rapid development cycles meant that users could see the results of requested changes very quickly, leading to increased user involvement at all stages and a corresponding high degree of user confidence and ownership of the implemented solution.

#### Techniques:

This project made extensive use of kinodb's capabilities. The following is a brief summary of the kinodb features that were used in the implementation of this system:

- Automatic creation of all user input forms. In a relatively complex system (with dozens of major entities) the design and management of user input forms is a major task, and makes the iterative development of systems prohibitively time-consuming, error-prone and expensive as each change requires



reworking of previously developed input forms, wasting much effort and introducing faults into previously working code. kinodb dynamically generates each input forms as it is required.

- kinodb's SQL Generation Engine removed any need for the coding of complex SQL statements. Again, this is a major enabler of iterative systems development as changes of any magnitude can quickly be made without the need to recode and retest thousands of lines of SQL code.
- Rigorous maintenance of data normalisation. Any complex data design is compromised if data is denormalised. kinodb's ability to transparently manage the referential integrity of complex relational data sets allows the designer to concentrate on the design of the database without being concerned with the complexity of the SQL that will be required to implement and manage it.
- Automated maintenance of hierarchies of types was used, enabling the inheritance of asset type characteristics (attributes, relationships etc.) down the hierarchy of asset types. The ability to transparently maintain these hierarchies enables the implementation of a relational database that exhibits characteristics more normally associated with object-oriented technologies.
- Fast generation of HTML reports for web deployment. As kinodb fully understands the structure of the database, creating complex reports from the data model is simple.
- Support for Binary Large Objects (BLOBS) was used in this system to enable the storage (and automated resizing, thumbnail generation and web distribution) of data collection guidance diagrams and photographs of sample assets.