

Project: GlaxoSmithKline Service Oriented Architecture Repository (SOAR)

GlaxoSmithKline's R&D IT developers have produced a number of programming modules that address various tasks relevant to the ongoing work of the company. There is virtually no sharing of the intelligence about these modules: the fact that they exist, where they are located, or how they can be integrated into a new project. There is one intranet site that documents Web Services, but does not register how the services relate to the business.

The vision of this project is to provide an indexing service into all service-oriented modules, including:

- Web Services
- Message Queues
- Data Stores
- Applications

To help relate the modules to the business, they are indexed according to the Information Object they act upon, and the type of action (Produce, Consume, Provide or Transform). The types of Information Objects are drawn from a company-wide taxonomy known as the GSK Thesaurus.

Data Structure and Interface

Prior to presenting a full proposal for the project, it was necessary to generate interest by presenting the concept to the R&D IT Executive Committee. A small, quickly developed, working prototype was key to generating this interest (the idea had been presented verbally a couple of times, but "they didn't get it".)

The quickest path to a network hosted, working prototype was to use PHP and MySQL. Initial design of the data structure only needed to support selected nodes in the GSK Thesaurus and most of the fields exposed in the interface. A single table with 8 columns sufficed:

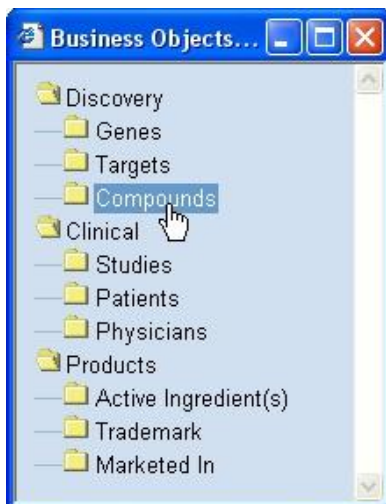
tree_nodes

Field	Type	Attributes	Null	Default	Extra
<u>node_id</u>	int(11)		No		auto_increment
node_name	varchar(48)		No	' '	
node_parent	int(11)		No	0	
node_type	varchar(30)		No	' '	
component_type	varchar(30)		No	Web Service	
node_description	varchar(128)		No	' '	
node_status	varchar(30)		No	Development	
node_support	varchar(30)		No	None	

Indexes :

Keyname	Type	Cardinality	Field
PRIMARY	PRIMARY	25	node_id

Since the purpose of the data structure is to support the prototype interface, each entry is a node in the tree. The node types can be either a Folder or one of the enumerated types (Web Services, Message Queues, Data Stores or Applications.) The placement of a new entry in the tree is determined by selecting its node_parent (the Folder it belongs in.):



The database and the “finished” interface of the prototype (below) will ultimately require more fields – identifying the location of the module, its owner, and the procedures involved in integrating it.

