A New Database Engine for Engineering/Manufacturing and Geographic Information Systems

Linguistic Technology System (LTS) has designed a novel database/cloud computing framework that is fully optimized for the requirements of back-end servers supporting native/desktop applications (applications which are run as standalone programs rather than viewed through a web browser). This framework can be used to facilitate the implementation of software applications which manage manufacturing and engineering workflows.

At the core of this technology is LTS's new Hypergraph Data Modeling (**HGDM**) protocol that uniquely combines property graphs with hypergraphs.

Software Development Challenges which are Addressed by LTS Tools

- Software must integrate heterogeneous data in a manner that is easy to visualize so that users can have all the data which they need on hand. Such difficulties become manifest when, for example, the visual display of an industrial part may need to be joined with a table-display providing information about the physical properties of that part, along with a key-value display providing logistical data such as the part number, dimensions, factory of origin, recommended lifetime for use, or guidelines for safe operation/installation. In general, software providing Computer Aided Design needs to combine interactive graphics with scientific, industrial, or commercial information about physical materials, parts, and products. Some of this information is conducive to structured displays such as spreadsheets and key/value pairs; other data requires visual displays such as charts, plots, or **3D** views. In addition, software applications often have to manage text and natural-language resources such as **PDF** files/manuals and product descriptions. LTS's **GUI** design method, which we call "VersatileUX," helps programmers create multi-faceted **GUI** displays for viewing such mixtures of diverse data structures.

- Profiles of real-world objects need to coordinate multiple forms of analyses to construct computational models via Digital Twins.[1] For example, the ability of a manufactured part to withstand external forces — a product both of how the part is constructed (its internal joints and weak points) as well as its materials of composition — can be approximated by simulating the part's exposure to physical and/or environmental factors, or it may be empirically tested using a physical prototype for the part in question. Test results also generate metadata concerning when/where the test was conducted, who observed the test, the degree of probability that the tests results are accurate representations of real-world performance, and so forth.[2] Collectively, test data and metadata define one facet of a Digital Twin, which then needs to be integrated with other facets such as **3D** displays and product/industrial information. LTS's tools provide data-integration so that these diverse kinds of data can be pooled into a common database for use by engineering/manufacturing applications.

---

[1] Digital Twins are used to represent physical products and Cyber-Physical devices for several reasons: to monitor physical attributes/settings in real time; to analyze physical properties, such as **3D** geometry, kinetics, and stress-mechanical behavior; and to prototype or provide information about physical parts, products, or devices. A single kind of physical object may be paired with a collection of distinct software components, each of which display, model, or analyze some facet of the object's properties.

[2] In the case of digital simulations, the results of simulated tests are likewise a facet of the digital twin, but so too are technical details of the simulations themselves. Integrating digital-twin data requires functionality such as ingesting new test results when they become available, maintaining a history of test results, and potentially initiating simulations in the case of digital/virtual testing.

LTS's Novel Database Architecture

- The LTS **HGDM** technology introduces novel features to aid in the implementation of industrial, manufacturing, and **GIS** software components. At the core of this technology is "**ConceptsDB**," a **C++** database engine which shares many features of HyperGraphDB but also supports the popular property-graph model.

- LTS combines these two different models of graph database engineering yielding a new "property-hypergraph" paradigm which LTS deploys in a **C++** environment, thereby bringing to **C++** both a property-graph and a hypergraph design which has previously been commercially implemented only in Java. Implementing property-graphs and hypergraphs in **C++** allows state-of-the-art graph database designs to be deployed in an environment that interoperates seamlessly with the large ecosystem of scientific, engineering, and Computer Aided Design applications and code libraries written in **C++**. [3]

- A similar architecture allows **HGDM** to be applied to problems in Geographic Information Systems, particularly the integration of geospatial data with **GIS** "objects." Geospatial imaging and mapping frameworks, such as the **ArcGIS Qt SDK**, have a system for associating generic object data to geographic locations and regions. This data can also be used as an annotation for satellite images, panoramic photography, and other images of natural/agricultural features. **GIS** data integration involves merging **GIS** and image data by lining up image coordinate space with geospatial coordinates (latitude/longitude and/or map axes) as well as associating geospatial locations with industrial/commercial data.[4] **HGDM**'s HyperGraph Type System can be used to effectively acquire and model information from a variety of industrial or scientific sources for injection into an underlying **GIS** information space.[5]

  In order to facilitate software development using **HGDM**, LTS provides a customizable development environment to aid programmers, in sectors such as manufacturing/engineering and scientific computing, who create software applications, plugins, and code libraries employing **HGDM**.[6] The LTS development environment includes a customizable suite of tools under development, including **ConceptsDB**, "Native-Cloud Native" or **NCN** (a cloud service library), "**THQL**" (a graph-database query language) and "dsC" (a dataset creator for scientific publications).

---

[3] **HGDM** incorporates and generalizes the idea of a Hypergraph Type System which was introduced by HyperGraphDB. In the context of HyperGraphDB, the Hypergraph Type System is designed to route information between applications and databases, but in **HGDM** a similar Type System is generalized to provide a wider range of data-integration capabilities. This can include routing information between **3D** graphics displays and modules for simulated testing; or between **PDF** documents (such as a catalog of parts for sale) and a parts database; or many other combinations of distinct digital-twin facets. Data-types registered with the **HGDM** Type System encapsulate functionality for exporting data to a persistent data store (such as a property-hypergraph database), sharing data via a cloud service, or constructing **GUI** components to visualize particular data structures in the form of dialog windows that can be embedded in desktop-style software applications. Each type within the **HGDM** Type serves as an intermediate connection point tying application-level data types with database, cloud networking, and **GUI** representations. This architecture simplifies the process of creating a database and/or a cloud-based back-end for software applications, as well as creating or extending applications' **GUI** front-ends.

[4] For example, energy companies may associate geographic locations with metrics such as the presence of subterranean oil or gas reserves, or calculations of wind or solar power which may be harvested in that location.

[5] This **GIS** development framework also has applications in fields such as government and public health, addressing concerns such as transportation optimization, epidemiology, and environmental monitoring.

[6] The standard version of this development environment includes popular code libraries used in physics and engineering for data analysis and visualization, including graphics and **3D** analytics engines such as Meshlab and **VXL**, physics/molecular applications such as IQmol and Paraview, and an assortment of mathematics, data persistence, and image-analysis libraries used across diverse fields in science and industry (including Eigen, Armadillo, Positlib, **DCMTK**, Common Workflow Language, TileDB, Cap'nProto, and more), along with libraries focused on internal **C++** programming requirements, such as data collections, generic programming, and enhancements to the **C++** type system. LTS's development tools combine this large suite of popular libraries into a central development platform, customizing the process of compiling and extending these libraries to promote cross-library interoperability and front-end **GUI** development. Compared to using the libraries in their traditional format, LTS's development tools make it easy for programmers to extend the libraries with data-export **GUI** features based on **HGDM**.