E-R Approach to Distributed Heterogeneous Database Systems for Integrated Manufacturing

Hemant K. Jain
University of Wisconsin-Milwaukee

Mohammed I. Bu-Hulaiga
King Fahd University of Petroleum & Mineral

A critical step towards computer integrated manufacturing (CIM) is the integration of computer-aided design (CAD) and computer-aided manufacturing (CAM). Well developed CAD, CAM, and Business data processing systems generally exist in an organization. These systems represent years of work and very large investment for the organization. Thus a necessary condition for integration is defining schemes through which existing engineering and non-engineering data can be represented and manipulated in an integrated way. In this paper a Common Data Model (CDM) approach to distributed heterogeneous database is adopted for CAD/CAM integration and an Extended-Entity-Category-Relationship (EECS) Model (an extension of ER model) is proposed. The use of EECS model as a common conceptual data model for representing engineering and non-engineering data is discussed.

Databases have been recognized by the CAD/CAM community as an ideal medium to integrate isolated “Islands of Automation” and to speed up communication among different design manufacturing and business related software packages. This can increase the consistency of the data and can avoid costly data conversion.

In a manufacturing organization different functions require different data types. Graphical and geometrical data are generated during drafting and engineering design process. These data are represented and stored using entities and data structures suitable for displaying graphical images or for constructing mathematical models of parts for further analysis. Engineering data are generated and used by analysis packages such as structural and thermodynamic analyses, simulation of motion, and material flow. Each analysis package has unique requirements for data input and output. Most manufacturing data takes the form of a procedural plan or sequence of actions. Textual data are found in all phases of integrated manufacturing from design through marketing; however, the heaviest use is in managerial applications such as production scheduling, cost estimation, quality control, inventory control, sales and marketing.

Thus in an integrated manufacturing environment the problem of heterogeneity has to be accepted and dealt with at the hardware and software levels. The strategies such as standardization and top-down system building even though relevant are not capable of resolving complexities raised by the simple fact that design-manufacturing and business data processing systems were available only from different hardware and software vendors. Thus existing CAD, CAM and Business data processing systems in an organization are generally heterogeneous in terms of hardware and software. These systems represent years of work and very large investment for the organization. However awkward the situation might be, the solution is in defining schemes through which existing engineering and non-engineering data can be represented and manipulated in an integrated way.

The concern here is to define a framework that maps data from one representation dictated by a specific DBMS to another. Clearly, the basic requirement is that the mapping should not affect the intrinsic value of the data. The prevalent approach for implementing this mapping is to define a common data model (CDM) at the global level(Bernhardt & Johnson, 1983). CDMs are used in...
Honeywell’s DDTS (Devor and Weeldreyer, 1980) and Computer Corporation of America’s MULTIBASE distributed architectures as global mapping schemes (Landers and Rosenberg, 1982). This paper presents an extended Entity-Category-Relationship model (an extension of E-R model), to define a conceptual schema for representing engineering data in a distributed, heterogeneous, design-manufacturing environment.

**Engineering Data Management**

The principal motive for considering engineering data (ED) as a self-contained category is the obvious inability of current DBMSs to effectively, let alone efficiently, accommodate design-manufacturing data. Engineering Data stems its differentiation from a number of well-celebrated qualities (Buchman, 1984; Koriba, 1983). They are generally complex, i.e., multiple entities are required to describe a basic object. This feature of engineering data calls for complex data structures capable of handling objects and their attributes. Another characteristic of engineering data is object composition. It indicates that a hierarchy of basic objects contribute to the object at the root of the tree. Additionally, more than one composite object is expected to use lower-level objects as building blocks (e.g., a certain curve could be in a number of composite objects).

The above two characteristics identify the lack of adequate data types to represent engineering data in existing database setting, hence, the rejection of off-the-shelf DBMSs as a means for managing engineering data. The deficiency in current DBMSs has resulted from the fact that present data models were conceived, designed, and implemented not particularly for the CAD/CAM environment. Thus, one may need to extend them (by introducing new more powerful data types), mix them (by using relational-network structures as in the IPIP (Bernhardt and Johnson, 1983)), or introduce totally new models that serve the strengthening trend of using computer-based systems in design and manufacturing.

Stonebraker, et al. suggested abstract data types and abstract indices (1983). Johnson et al. described a network-relational system to provide for the hierarchical nature of engineering data by viewing an engineering structure as a single entity called structured entity (1983). This architecture gracefully accommodates complexity and composition of engineering data. Lee and Fu designed a recursive and nondeterministic data model for handling engineering data (1983). Their approach is based on the aggregation and generalization concept of Smith and Smith (1977). This model is conceived to resolve issues related to complexity of engineering objects. Haynie (1981) proposed a network-relational hybrid data model. In their work, the relational model is used to store data regarding objects in the database; and network schema is used for managing composition of objects.

Recently the use object oriented model is advocated for CAD/CAM data bases. Cammarata (1986) developed an object oriented data model (ODM) for representing engineering data. A prototype implementation of the model was also developed. Kim et al. (1989) have built a prototype object-oriented database system called ORION which supports functions required by applications from CAD/CAM, artificial intelligence, and office information systems. The functions supported by ORION include change notification, composite objects, dynamic schema evolution, and transaction and query management. Object oriented approach has also been used for VLSI/CAD design (Gupta, et al., 1989).

The object oriented approaches and the implementation of DBMS’s based on this model works well in the situation where a new system is designed from scratch. They do not offer a solution to the problem of integrating existing heterogeneous systems.

The above citations, albeit incomplete, represent the general trend of either extending data models as a means for building DBMSs that are more suitable for handling engineering data and more efficient in satisfying their requirements or developing new data models. Detailed review of engineering data requirements can be found in Buchman, 1984.

**Distributed Database Architecture for Integrated Manufacturing**

Traditionally, Computer-Aided Design (CAD) and Computer-Aided Manufacturing (CAM) were treated as self-contained systems. Also the administrative systems supporting inventory control, purchasing, sales, etc. are generally developed and managed as separate systems. The rapid evolution of Computer-Integrated Manufacturing (CIM) Systems makes the coupling of these long-separated software systems essential (Halevi, 1980). The major obstacle to such integration is the theoretical and technological base that is necessary to guide its implementation. Also since existing CAD, CAM, and business systems represents years of work and large investment for the organization any approach to integrating them must protect this investment.

Distributed system technology can be used for linking various systems to accelerate the integration. In these systems communication network provide the physical means for connecting the dispersed systems serving different phases of manufacturing, and a heterogeneous distributed database management system provide an integrated view of the data. However, many issues related to data representation and mapping needs to be resolved. The concern here is to develop an approach for managing and integrating engineering data that reside at various sites managed by different DBMSs such that they can be collectively manipulated to satisfy global queries.
The necessity of a CDM to represent engineering data is justified by the peculiar nature of engineering data. This implies that CDM’s data structures must be designed with that peculiarity in mind. Some of the issues peculiar to representation and manipulation of engineering data are presented here.

In a typical engineering environment when a design is being created (e.g., an airplane wing) design groups working on different aspects of the project need distinct, separate copies of the design to carry on their share of the task independently within the framework of the baseline. As the execution of the task progresses there is a need for maintaining multiple versions of the design. Each version is related to its owner i.e., to the group concerned with a particular aspect of the project. Moreover, the versions need to be organized such that the history of the evolving baseline could be tracked when necessary.

Alternative methods have been proposed to support such an environment. IPIP (Bernhardt and Johnson, 1983) uses a mechanism for grouping data into data sets which are referenced by various users. This mechanism is the backbone of multi-version support, release mechanism, and archival of versions. The grouping mechanism is implemented in IPIP through the use of “ACCESS” and “STORE.” The “ACCESS” statement identifies the data sets that a user wants to work on. Consequently, the effects of the manipulation commands are confined to the defined data sets. Two types of versions are identified: major versions and minor versions. Major versions are full duplicates of current baselines. They are useful for obtaining baselines that have no minor versions linked to them. On the other hand, minor versions are made up of the modifications committed to major versions, and are linked to the major version by a pointer. A release mechanism is necessary to safeguard the quality of the data set. This technique is conceived mainly to put restrictions on updating already released (i.e., finally approved and processed) data sets. To grant update rights, the user has to go through the appropriate channels. The reason for this is that released versions are usually arrived at after various discussions, and compromises from all concerned parties. Chou and Kim (1988) implemented a Versions and Change Notification system based on transient and working versions. The transient versions can be updated and deleted by the user who created it. The working version is considered stable and cannot be updated, but it can be deleted by its owner. Also transient versions can be derived from a working version.

The second requirement of common data model is evaluated at data and program level. At data level the schemas of local DBMSs should have equivalent representation in common data model. At the same level, the issue of converting data from one representation to an equivalent another one is treated. At the program level the issue of verifying the equivalence of global and local DML commands is considered.

Finally, the third requirement of common data

---

**Figure 1: Distributed DBMS Architecture for Integrated CAD/CAM**

Figure 1 represents a distributed database architecture for integrated manufacturing. A global schema representing all the data is defined. Centralized database design principals can be used for the design of global database schema. However, the data model used at this level must be capable of representing engineering and non-engineering data and should be able to analyze and map global queries to local DBMSs. In a heterogeneous environment, the most convenient way of satisfying the general mapping requirement is to adopt a layer of Common Data Model (CDM) and Data Manipulation Language (DML). This layer must be general enough to communicate efficiently with all different DBMSs in the system. Thus, mapping needs to be defined between each of the local DBMSs, and CDM and DML. For this purpose a layer of local mapping schema can be added to each of the local DBMSs. Thus, this architecture protects the investment in the existing systems, and maintains the autonomy of local DBMSs. Local queries can be efficiently answered directly by local DBMSs.

**Requirements of the CDM**

The requirements of the Common Data Model and data manipulation language to be used in the distributed DBMS architecture for integrated manufacturing can be summarized as:

1. Capability of representing engineering data and textual data.
2. Suitability for translating data and queries from and to local DBMS representations.
3. Suitability for representing schemas at global and local levels.

The necessity of a CDM to represent engineering data and textual data is justified by the peculiar nature of engineering data.
model is related to the efficiency of the model, in represent-
ing local schemas such that global queries can be processed
efficiently. In addition, the designer must, somehow, re-
solve the issue of conflicts such as name, scale, structural,
and abstraction conflicts that usually arise between schemas
of CDM and local databases. The worst aspect of dealing
with these conflicts is that there is little that can be done by
the designer, since the bottom-up approach is, by far, the
dominating development methodology. Nevertheless, an
auxiliary database can be implemented, if the designer
chooses not to resolve all conflicts in the common data
model (Navathe and Gadgil, 1982).

**Heterogeneous Distributed Databases**

The current heterogeneous distributed database technology is represented by a number of prototypes. Some
of the widely reported prototype systems are: MULTIBASE
developed by Computer Corporation of America(Landers
and Rosenberg, 1982), heterogeneous SIRIUS-DELTA de-
signed by INRIA of France (Ferrier and Stangret, 1983), and
Distributed Database Test System (DDTS) a prototype ini-
tiated by Honeywell (Devor and Weeldreyer, 1980). The
first two systems are briefly discussed here. The DDTS is
reviewed here in detail to assess the implication of using an
approach similar to its Entity-Category-Relationship model
for representing the common data model.

MULTIBASE is a distributed retrieval system. It
has two basic components: global data manager, and local
data interface. The former is responsible for global aspects
of a query, while the latter’s function is to interface DBMSs
at various sites. The system has a four schema architecture:
global, local, auxiliary (used to solve conflicts), and local
host schemas. The global data model and DML supports the
notion of entities, functions, and generalization hierarchies.
MULTIBASE does not support update applications, they are
processed under the direct responsibility of local DBMSs
and are not globally coordinated. Likewise, no facility is
provided to synchronize read operations across several sites.
The major objective of MULTIBASE is to interface pre-
existing systems without modifying their software. No
specific facilities have been provided for versioning, update
propagation, and handling of variable length unstructured
data required for Integrated Manufacturing. These features
can probably be implemented due to the open architecture of
the system.

The SIRIUS-DELTA has a four layer structure.
The highest level layer is the ‘SIRIUS-DELTA’ DBMS.
The second level SILOE layer handles functions related to
query optimization. The third layer, SCORE, provides
concurrency control and reliability. Finally, SER layer man-
ages the distributed execution of applications. SIRIUS-
DELTA deals with heterogeneity and homogeneity alike.
For a site to be included in the distributed system, a number
of common functions need to be defined by the local DBMS.

These functions are described using protocols provided by
the SILOE, SCORE, and SER layers. Collectively, the
common functions represent a pivot system. These func-
tions are related to concurrency control and locking.

The DDTS system is based on the five schema architec-
ture proposed by Devor et al.(1982). Figure 2 provides a schematical description of the DDTS architec-
ture. The system has two main logical processors: applica-
tion processors (AP) and the data processors (DP). The APs
main function is the translation of the external schema to the
conceptual schema. Some of the other functions are enforce-
ment of integrity constraints, selection of the sites for trans-
action execution, and coordination of transaction execution.
Each data site of the distributed system has a Data Processor.
It carries out a number of functions such as: mapping to local
internal schema, and optimizing access operations to local
database. Local DBMS executes operations specified by
the mapping function.

The DDTS uses an extension of the Entity-Rela-
tionship model (Chen, 1976) to implement the global/local
representation schemas. The extension is known as the
Entity-Category-Relationship (ECR) model. A category
defines the role entities play in relationships with other
entities in the data base. This concept is similar to the one
introduced by Smith (1977). However, in the ECR model,
the emphasis is on the role that might be played by the entity
types. A number of examples of this model can be found in
Devor, et al., 1982. The DML used for specifying queries is
GORDAS (Graph-ORiented DAta Selection) (Elmasri and
Widerhold, 1981). Currently, DDTS is capable of integrat-
ing CODASYL-type DBMSs.

![Five-Schema Architecture](image-url)
Common Data Model for Integrated Manufacturing

In a previous section, requirements of the common data model to support distributed database for integrated manufacturing system were discussed. The concern here is to use those requirements in specifying a Common Data Model (CDM) capable of supporting engineering as well as non-engineering data. Also, we need to specify a Data Manipulation Language to complement the common data model.

The major requirements for CDM are listed here for easy reference:
1. capability for representing engineering and textual data.
2. suitability for translating data and queries.
3. suitability for representing schemas at global and local levels.

We are not aware of any models that satisfy all of the above three requirements. While DDTS is closest to satisfying CDM requirements, it falls short of achieving the first requirement in the list above. The model presented here to resolve this problem is the extended entity-category-relationship model (ECR). It is derived from two distinct extensions to the entity-relationship model; namely, the entity-category-relationship model (ECR) (Weeldreyer, 1980) and the extended entity-relationship model (EE-R) (Neumann, 1983).

E-R Model

In E-R model, information is modeled in terms of entities and relationships between entities. The real world objects, events, concepts, and phenomena are represented as entities. It is an abstraction of objects and facts as they exist in our mind. The entities are represented as a collection of values. They represent selected properties of the modeled object. To identify which value represents which property, the concept of attribute has been introduced as a function which maps an entity set into a value set. An entity set can be described as a class of entities that can be described by the same attribute set.

The relationships between real world objects can be expressed as relationships between entities. These can be modeled as association of values that identify entities which are related.

The Entity-Category-Relationship (ECR) Model

The ECR model extends the ER model in several ways. In particular, it classified the entities in two ways: (1) according to similarity of properties, (2) according to the roles an entity might play in a relationship. This results in the ability to treat complex and seemingly higher-order relationships such as generalization [29] using a first-order logic of little more complexity than the relational calculus. Figure 3 presents an example of a simple database represented in ECR Model. The example, adopted from Weeldreyer (1982) shows the entities, relationships, and the type participation of entities in different relationships. Entities are classified into categories based on the roles they may play within relationships. For example, members of the automobile type and of the truck type may be categorized as vehicles in an owner-vehicle relationship. Similarly, members of the person type and of the corporation type may be categorized as owner in the same relationship. This concept of a higher level entity is the foundation of many object oriented databases (Cammarata, 1986).

The ECR model has obvious advantages. It supports mapping of ECR schemas to network and relational representations. It also provides necessary semantics for representing complex relationships. Finally, the ECR model is fully defined using what Weeldreyer calls “ECR Calculus,” which is a collection of first-order predicates (1980). It is important to note that even though the ECR model extends the E-R model in a variety of ways especially as far as model semantics are concerned, the model has been primarily designed to handle textual data. To eliminate this limitation we searched for a model which can supplement the ECR model, such that it can handle engineering data as well. We confined our search to E-R based models. The motivation for this was that they are the foundation of most object oriented databases used for CAD/CAM databases, additionally developed techniques exist for mapping E-R based schemas to schemas represented in popular data models such as: relational and network model (Cardenas and Pirahesh, 1983).
1980; Chu and To, 1980; Hwang and Dayal, 1981). This is necessary to satisfy the second requirement of common data model. Neumann’s (1983) Extended-Entity-Relationship (EE-R) model which is described below was found to be a good match.

**Extended Entity-Relationship Model**

EE-R (Extended Entity-Relationship) model has been proposed by Neumann (1983) to handle hierarchical entities. He defined a new class of entities called kernel, and a new relationship class named reference class.

A composite object (i.e., an object with a substructure) is represented by a kernel entity. Use of reference relationship is confined to entities of the kernel class. In addition, Neumann included in his model owned and regular entity classes, and weak and regular relationships which resemble those of the E-R model (Chen, 1976). Thus EE-R model can represent the variety of roles an entity plays in an integrated manufacturing environment. An extended Entity-Category-Relationship (EECR) Model which can represent the textual and engineering data is presented here.

**Extended Entity-Category-Relationship (EECR) Model**

Although it is possible to use the framework of E-R, ECR, and EE-R model to represent engineering objects, none of these models alone can express the complexities of integrated manufacturing environment explicitly. As an example consider the description of an object’s shape. It can be described by a variable number of faces, edges and vertices. Some objects require a large and variable number of entities to represent them. These objects are termed as complex objects. Other objects requiring single entity to represent are termed as simple objects. An engineering object is often composed of a number of simple and/or complex objects. Such an object is termed as a composite object. A composite object needs to be identifiable as a single object. Some of the reasons for this requirement are: (1) it should be possible to manipulate such an object as a whole, (2) in a multi-user system, the whole object can be locked by a single lock rather than locking each individual entity.

Additionally an engineering object can be described from several points of view. For example, an electrical circuit can be represented by its functional description, by its circuit diagram, and by its layout. Similarly for CAD/CAM design data, apart form graphical and geometrical representation of parts discussed earlier a comprehensive and user oriented model must represent entities such as holes, slots, cutouts, and flanges and the role these entities play for different users; e.g., a hole indicates a path of electrical wires for electricians; while to a thermodynamic engineer, a hole means a source of heat loss; and to a process planner, a hole means a drilling operation.

The EECR model proposed here defines four entity classes and four types of relationships. The entity classes are kernel, owned, regular, and category. The relationship classes are reference, weak, limited, and complete. The above entity and relationship classes are described here with the help of an example.

**Entity classes:** A regular entity is a representation of non-hierarchical objects, concepts, events and has properties which describe it. Entities which have similar properties are classified as regular entity set. For example, employees, inventory, sales, purchase order, etc. in manufacturing environment can be represented as regular entity. This will be represented as a regular rectangle (shown in Figure 4).

As described before a category classifies the entities based on the role they play in the relationships. For example, automobiles and motor homes can be represented by a category Vehicle, because they play the same role for registration. Categories are not necessarily disjoint so that a given entity may be a member of several different categories. Also, it is possible that a category may contain entities of only one entity set.

Kernel entity is an entity set used to identify a composite object that can be referenced as a unit and might be used as a building block in more complex structures. Thus, a composite object will be represented by an entity from Kernel class. A rectangle with two vertical lines represents a kernel entity.

Owned entity is used to provide the hierarchical
structure peculiar to engineering objects. The existence of these entities depends on the existence of a higher level entity. They characterize or describe the high level entity and are owned by the higher level entity class. In an engineering environment the kernel entity represents the independent composite objects, while the component parts of this composite object is represented by owned entities.

**Relationships**
The relationship is defined as a mathematical relation, i.e. a subset of a Cartesian product of two or more entity sets. Depending on the entity sets participating in the relationship following types of relationships are defined.

A weak relationship represents 1:n binary relationship between a kernel and owned entities or between two owned entities. In this relationship the existence of subordinate entities (n side of relationship) depends on the existence of higher level entity. This relationship can be used for representing components of engineering objects or various features of a part.

A reference relationship is used to relate a set of kernel entities. This class of m:n relationship can be used to connect a simpler composite object to one or more higher level composite objects. The simpler object can, however, exist in the database independent of any other objects.

A limited relationship is used for representing 1:n relationship in cases where a category or a regular entity is at the top level of the hierarchical structure. This relationship class is used for representing connections from category or regular entities to other entity classes.

A complete relationship is the most general of all classes. This n:m relationship relates entities of kernel, regular, and category classes as necessary.

For the purpose of illustrating the EECR model let us consider various models of cars being manufactured by an automobile company. Each model has many dealers and a dealer in turn can sell many models. This is represented by regular entities and complete relationship. The car has an engine (represented by Kernel Class entity) and engine consists of composite components such as piston, valve, etc. Each of these components has various features such as holes, slots, etc. represented by owned entity. For the purpose of production schedule all the components can be looked as a category PARTS. Figure 5 represents the example EECR model schema. Only few entities and relationships are shown here for illustrative purpose.

The above description of the components of the EECR model and the illustration represents some of the capabilities of this E-R extension which can be used for conceptual representation of CAD/CAM data in a distributed environment. In this environment the global schema will contain a comprehensive definition of all engineering as well as regular entities in all the databases. This is achieved without compromising the integrity of the database. Subsets of the global schema can be stored at the sites to represent the CDM level conceptual schema of the data resident at that particular site. When a global query is presented to the system it would be distributed and optimized. The subqueries resulting from the distribution process are transmitted to relevant sites. The mapping mechanism translates these subqueries to calls understood by the local DBMSs.

Thus, EECR model satisfies all CDM requirements discussed earlier.

**Common Data Manipulation Language**
The basic requirements of a DML in a distributed environment is its ability to communicate with the internal structure of the database in an efficient way. In addition to the traditional requirements of data manipulation and query management, a DML in a CAD/CAM system needs to support efficient schemes for concurrency control, versioning, and archiving. The EECR model proposed here can be used with an E-R based data manipulation language e.g. GORDAS (Elmasri and Widerhold, 1981). The modified GORDAS needs to have traditional DML capabilities in addition to feature peculiar to the extension described above.

GORDAS is an integral graph-oriented non-procedural language. It selects the desired data without manipulating the underlying database. The most attractive feature of this DML, as far as this work is concerned, is the fact that is has been specified and tailored to the E-R model. Queries stated in GORDAS have two clauses: GET and WHERE. The GET clause specifies what information is to be re-

*Figure 5: An Example of EECR Model Schema*
trieved. The WHERE clause specifies the conditions under which the information is to be retrieved. A sample query might clarify the usage of the language.

GET NAME OF EMPLOYEE
NAME OF DEPARTMENT
WHERE (Age of employee = 25) and (CODE of department = 123)

The language has a number of aggregate functions, such as SUM, AVG, SD, MAX [ ], and MIN [ ]. GORDAS is classified as a graph calculus language because it selects desired information without manipulating the underlying database. Furthermore, the language achieves its manipulation via set formation and comparison. Set formation is represented in the language by UNION, INTERSECT, DIFFERENCE, while set comparison by the powerful operation of INCLUDES.

A principal motivation for using GORDAS (in addition to its E-R connection) is its support of hierarchical structures. However, the language needs to be enhanced to enforce the constraint such as owned entities within the kernel structure may not be referenced directly as independent entities.

Conclusion

The problem of managing engineering data in a heterogeneous distributed environment is still a challenge. This paper presents a distributed system architecture for integrating existing CAD, CAM, and Business data processing systems. A Common Data Model (CDM) approach has been used, and an extended Entity-Category-Relationship (EECR) model has been proposed. The EECR model is capable of representing engineering and non-engineering data and can serve as CDM. Also the use of extended GORDAS, an E-R based DML, has been suggested to serve as a global DML. Detailed specification of data description language for EECR model and the specification of extended GORDAS are currently being developed. Also techniques for mapping EECR model to popular relational, hierarchical, and network models are under development.

References


Hemant K. Jain is an associate research professor of MIS in the School of Business Administration at the University of Wisconsin-Milwaukee. His research interests lie in the areas of distributed system design, database management, decision support systems, model management and expert systems.

Mohammed I. Bu-Hulaiga is an assistant professor in the College of Industrial Management at King Fahd University of Petroleum and Minerals, Saudi Arabia. His research interests include database management, model management, and expert systems.
Related Content

Development of Interactive Web Sites to Enhance Police/Community Relations
www.irma-international.org/chapter/development-interactive-web-sites-enhance/18565/

Clustering Vertices in Weighted Graphs
Derry Tanti Wijaya and Stephane Bressan (2012). Graph Data Management: Techniques and Applications (pp. 285-298).
www.irma-international.org/chapter/clustering-vertices-weighted-graphs/58615/

An Overview on Signature File Techniques
www.irma-international.org/chapter/overview-signature-file-techniques/20750/

Supporting the Full BPM Life-Cycle Using Process Mining and Intelligent Redesign
www.irma-international.org/chapter/supporting-full-bpm-life-cycle/7022/

Databases Modeling of Engineering Information
Z.M. Ma (2009). Selected Readings on Database Technologies and Applications (pp. 65-84).
www.irma-international.org/chapter/databases-modeling-engineering-information/28546/