A Support System for Design and Routing Plan

Hwa Gyoo Park, Ju Chan Shon, Sung Gin Park, Jong Myung Baik

Electronics and Telecommunications Research Institute
161 Ga Jung -Dong, Yusung-Gu,
Taejon, 305-350, KOREA

Abstract

In this paper, we demonstrate the implementation case using component based development tool under development process for application developments. The tool suggested provides the programming environment for the development of distributed manufacturing applications primarily. The development tool is classified into visual component, logic component, data component, knowledge component, neural net component, and service component which is a core component for the support component edit and execution. We applied the tool to the domain of the design and routing plan to retrieve existing similar design models in database, initiate a model, generate a process plan, and store the new model in the database automatically. Utilizing the tool, it integrates a geometric modeler, engineering/manufacturing database, and knowledge sources over the Internet.

Overview

It is becoming less possible for developers of integrated software solutions to quickly, efficiently, and inexpensively develop all of the functions and modules desired by actual users in the industries. The requirements can be achieved by reusing existing software components. The principle of component software is already changing the software development concept. The development tool is the proposed system's programming environment for development of manufacturing-wide 3-tier solution. It provides the entire system implementation cycle with tools for modeling, programming in the 4GL language, definition of data and table structures, and the design of graphical user interfaces. As a supplement to the development tool, it delivers a template library for the design and manufacturing and components that can be incorporated into custom functions. Figure 1 illustrates the total view of the proposed system development environment.

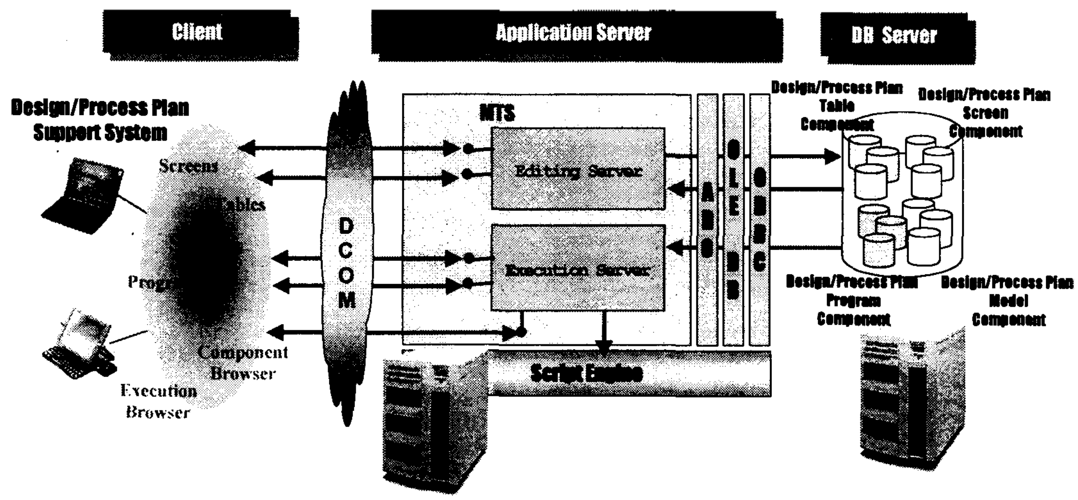


Figure 1 Proposed Development Environment

Framework of Development Tool

Structure of Development Tool

The principle of component technology making possible that standard engineering software use modules from other manufacturers. In the tool, the components are classified into 5 types.

(1) BCO (Business COmponent)
The BCO contains the SCOs, business application components, and business transaction components, however, in a narrower perspectives, only VCO, LCO, DCO, KCO, NCO are enclosed.

- VCO(Visual COmponent): The VCO encapsulates the visual user interfaces related a Screen ScreenProcess, Control and a Dialog component type. The ScreenProcess is broken down into ASP(After ScreenProcess), BSP(Before ScreenProcess), and CSP(Common ScreenProcess) once again. Transactions in the tool contains several screen or dialog steps, and the screen begins with the preparation and subsequent transmission of the input screen to the user's screen. The two type of screen step processing are ASP, and BSP.
- LCO(Logic COmponent): The LCO stands for the programs performing application logic, and upper-level method.
- DCO(Data COmponent): The DCO stands for the relational table components for applications, and element components.

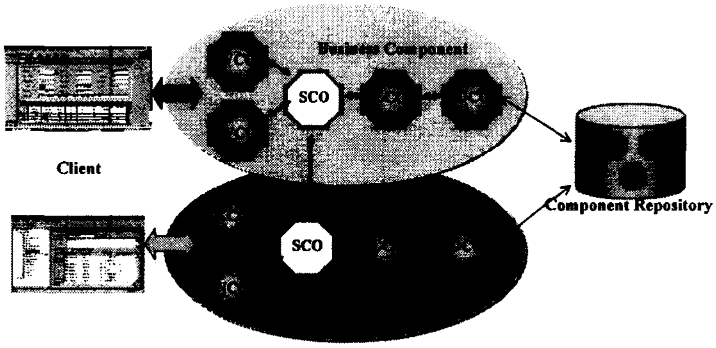


Figure 2 Proposed Development Tool Structure

- KCO(Knowledge COmponent): KCO encapsulates rules, meta-rule, and reasoning engine.
  - Knowledge Source Sub Component: it stores the facts and heuristics of domain experts.
  - Working Memory Sub Component: it contains the information the system has received on the problem at hand. Any information the component drives on the problem is stored in here.
  - Inference Engine Sub Component: it provides the system control and applies the expert domain knowledge to present situation.
- NCO(Neural net COmponent): KCO encapsulates Conversion Module, NN Database interface, NN Recall Module, NN Learning Module, Restructuring Module
  - Conversion Sub Component: it is a Sub Component for the conversion process.
  - NN Database interface: it is a Sub Component that stores and manages the coefficients, weighting vector, size, and learning patterns of neural network.
  - NN Recall Sub Component: it is a Sub Component that restructures neural network model utilizing the coefficients, weighting vector, and size stored in NN database.
  - NN Learning Sub Component: After reading the size and learning patterns of neural network in NN database, constructing a multi-layer perceptron model of neural network, and completing training process as specified, this Sub Component returns coefficients and weighting vectors to the NN database.
  - Restructuring Sub Component: A Sub Component that generates topological information.

Figure 2 illustrates the Proposed Development Tool Structure, which is consists of the briefly mentioned components.

(2) SCO(Service COmponent)
The service component is for the support component edit and execution. The SCO is a core component, which have a role of supporting component framework. It is a standardized interface for linking components, and services are an elementary part of open programming environments. The component is implemented to manipulate the components, attributes and methods. It acts as suggested system servers. Figure 3 shows us the system layers of infra-structure, framework, component, and applications.

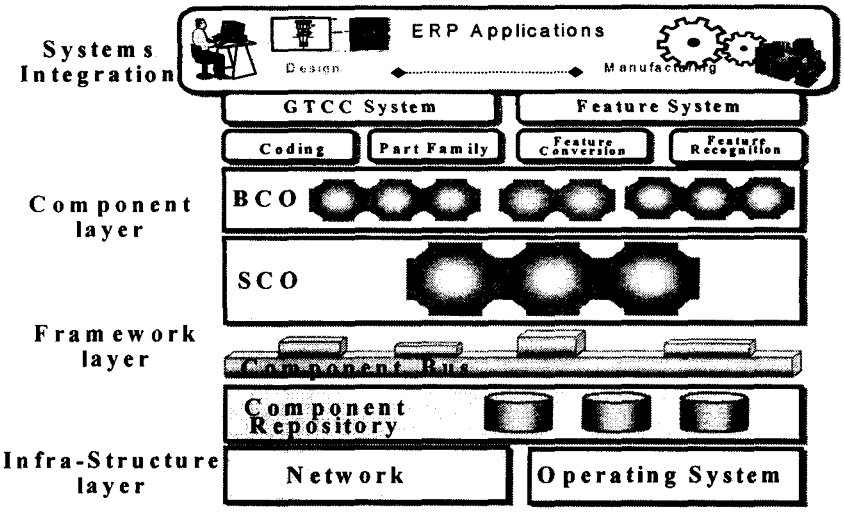


Figure 3 Illustration of Implementation System Layers

Generating Tools

For the implementation of application system, the system provides 4 types of editor and a browser to the developer. Two of them, the Program Generator and run-time Browser of the tool is shown as examples in Figure 4.

(1) Main Generator: it has the functions managing overall project and executing and controlling the various of editors. The editor can show bird's eye view of a project in progresses.
(2) Database Generator: It is a tool for constructing physical tables, and it manages the repository containing all the screen and business logic applied to the applications.
(3) Program Generator: It is an editing tool for the programming of a application domain, using VB script.
(4) Form Generator: it is a screen form editor for user interface. To make equal effects in the executed editors, ActiveX Control is utilized.
(5) Run-Time Browser: it is a browser to execute the actual application domain logic. The browser has the function displaying the data for screens, and the business logic is performed in the server.

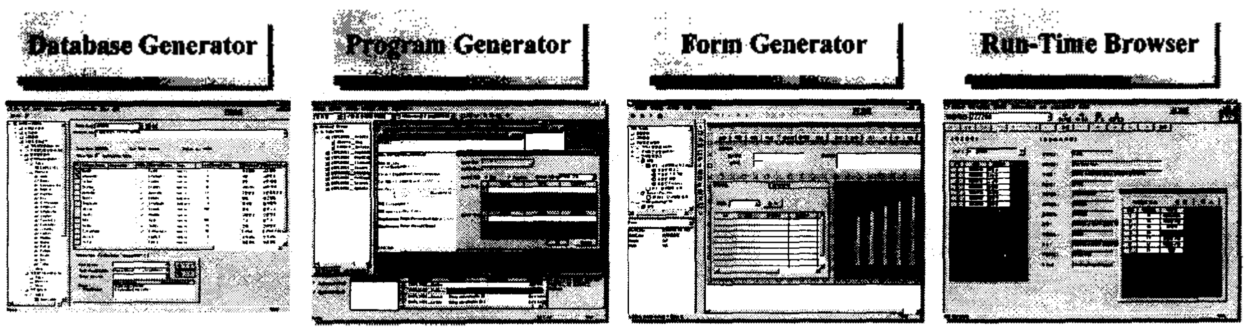


Figure 4. Type of Generator

Design and Process Plan Support

Overview of Applications

The implemented system provides a paradigm for design and process plan support as it combines the variant and generative approaches and improves the effectiveness of the CAD design engineer as well as process planner by applying old experiences to the planning problems anew. Figure 5 illustrates the whole system overview.

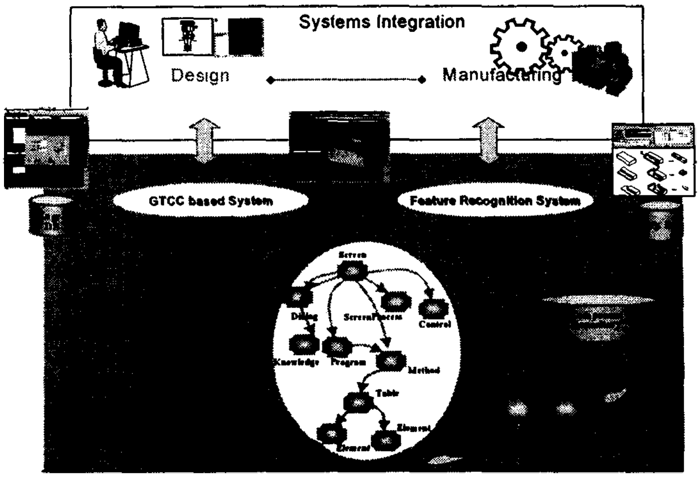


Figure 5. Application System Overview.

Part Family Based System

The group technology coding and classification (GTCC) encoding of a part is determined by user's response to queries provided by the system. The steps needed to generate a GT code and create a CAD model are described in more details. Figure 6 illustrates a use case of the manufacturing support functions for variant type of process plan. For this representation, a commercial CASE tool, SELECT, is utilized to analyze and design the system.

(1) design code

The system queries about the external and internal shape of a new part. Answers determine primary design attributes. The first query is used to determine part class: rotational or non-rotational. The queries depend on this answer and continue until all digits have been assigned.

(2) design template

If the CAD template model does not exist, the database retrieving procedure is instantiated instead. The template model is represented by a parametric file and consists of the primitives of a CAD model such as POINT, LINE, and CIRCLE, with unassigned variables such as x, y, and z coordinates for a point.

(3) manufacturing and process plan code

The system queries about some of the attributes that are useful to manufacturing, such as *dimensions, material, starting raw* workpiece shape, and accuracy. These queries are independent on the design code and the part family code.

(4) part data file

If the query fails to locate the template model, the system attempts to find a similar model. This method allows a user to modify a CAD model, as an alternative to the retrieval of an existing variant form.

The CAD model is modified using CAD interactive functions. After modifying the CAD model, a user must capture a picture of the CAD model in a bitmap format. This involves sizing and framing the model, saving it, and copying it into knowledge source.

(5) process plan template and generating process plan file.

Using manufacturing part family, user retrieves a template or similar process plan and modifies the plan. Finally, user reenters the system and updates the part database for two types of part families, form code, supplementary code, PP code, and part specification information about the new part.

(6) Update data

The system allows a user to modify the part database part file by executing a part database program which consists of a main program and 3 subprograms. A user can modify the name of the part (which has same name as the CAD model file and the model picture file) and the description of the part.

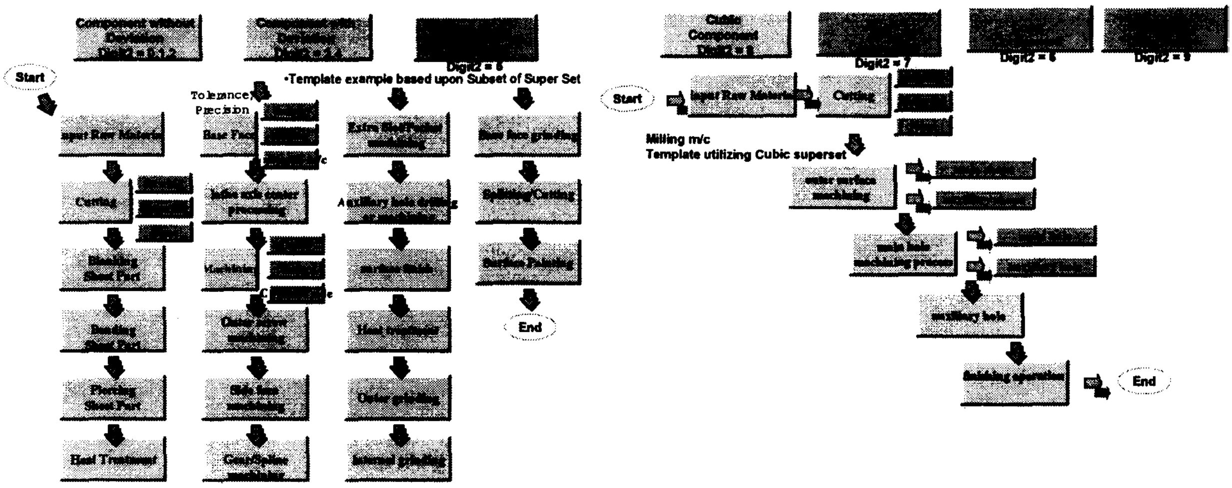
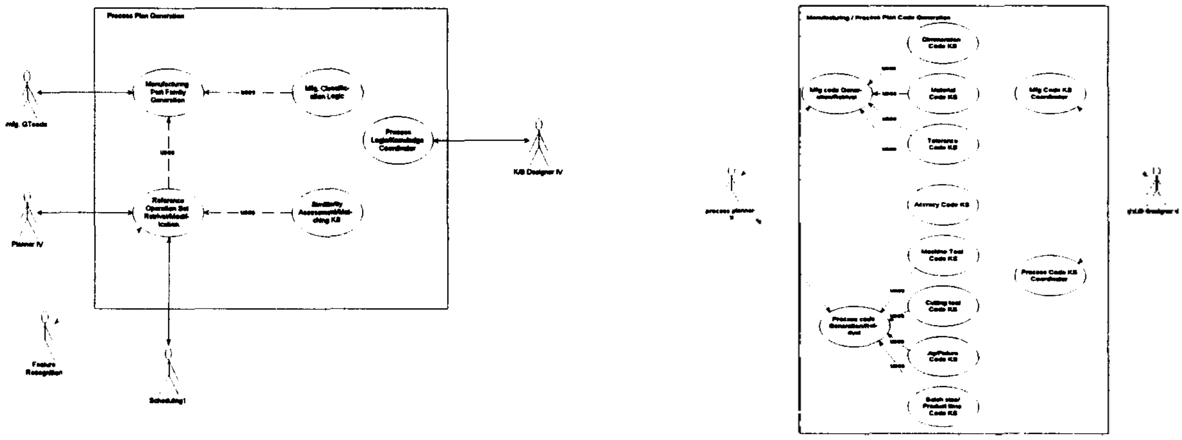


Figure 6. Process Plan Templates Example for Rotational and Prismatic Part



(a) Code Generation (b) Plan Retrieval

Figure 7. Partial Use Case Diagrams of Variant Process Plan

Feature Recognition System

For newly ordered parts, the system also provides a function of automatically extracting and recognizing features to convert design features into manufacturing features for the generative process plan. Figure 7 shows when the recognition system is initialized.

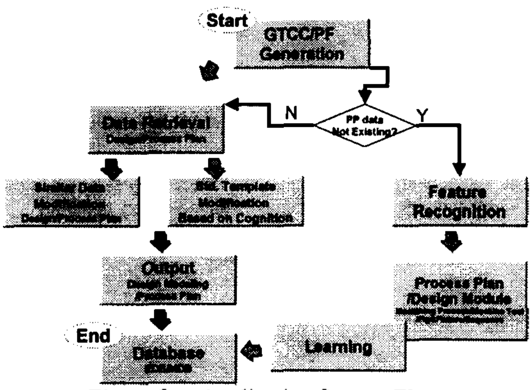


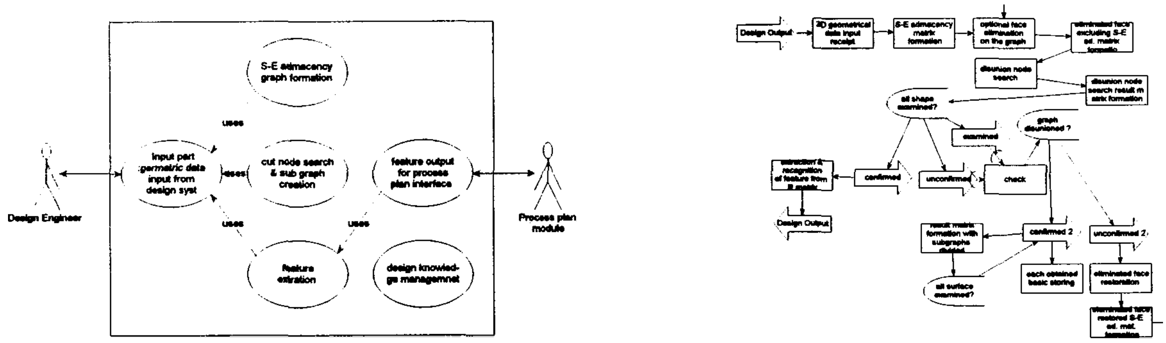
Figure 8. Application System Flow

The feature recognition function based on a back propagation model is instantiated when there is no history process plan or proper standard process in the database and the template library. For this, the part is conversed into manufacturing feature from the design feature firstly, and then process planning logic is initiated and matched into proper manufacturing resources to produce a generative process planning. Additionally, to make the system intelligent, the plan is generalized and adapted into the knowledge source. In this paper, we limit to the description of feature recognition processes.

(1) Extraction of Atomic Feature

Figure 9 illustrates a use case diagram for extracting feature module from three-dimensional geometric data under the present research. As illustrated, it comprises:

- as input part which receives input of geometric data from a CAD system,
- a processing part which forms a adjacency graph by extracting a surface and an edge from the geometric data inputted through the input part and extracts features by searching for a disunion node and creating subgraphs,
- a feature output part which outputs the feature required for a process planing extracted from the processing part, and a design knowledge management part which maintain structure of the module constructed on neural net and rule based knowledge sources. The feature output is interfaced with process plan module.



(a) Use Case Diagram (b) Thread Diagram

Figure 9. Use Case Diagram and Thread Diagram of Design Interface

(2) Surface-Edge Graph

The input part comprises an adjacency graph forming part which forms a adjacency graph for the surface and edge, a subgraph creating part which searches for the disunion node, and creates the subgraphs by using the surface-edge adjacency graph and features extracting part which extracts the feature by the surface-edge connection graph. The each individual subgraph detachable by a disunion node stands for an atomic feature. Figure 9(b), a process thread diagram, illustrates this process.

If there are two disconnected subgraphs by eliminating the optional surface from the matrix and after checking the connection of the remaining surfaces, this surface is fixed upon as a disunion node. Subsequently, it performs a step of forming the result of disunion node search into a matrix, a step of judging if all shapes are examined and a step of judging if the graph is divided into the two graphs in the case where all shapes are not examined. When the graph is divided into the two graphs, the present eliminated surface is a disunion node, and it performs a step of forming a result matrix with subgraphs divided by the disunion node. In the case where the graph is not divided into the two graphs and the result matrix is formed by subgraphs, it performs a step of judging if all surfaces in the graph under review are examined. If all surfaces are examined in the corresponding graph, it performs a step of storing each obtained basic shape and returns to the step of forming said surface-edge adjacency relation matrix. When all surfaces are not examined, it performs a step of restoring the eliminated surface, and repeats the step of searching for a disunion node by performing the step of eliminating the optional surface on the graph after the eliminated surface performs a step of forming a restored surface-edge adjacency relation matrix.

When examination of all shapes is finished as a result of performing the step of judging if all shapes are examined, it performs a step of extracting features from the result matrix and recognizing it. Thus, the present research enables a computer to work out a process planing automatically by providing feature information required for the process plan. On the other hand, the method of extracting features by means of such a disunion node is subject to restriction in the case where incidence surfaces for the feature are plural like a through hole. In order to bring a solution to it, the conception of said single disunion node is expanded into the conception of multi-disunion node. In other words, the feature finally separated from the above disunion node conception is a feature having one incidence surface and it can have several incidence surfaces again in the inside, so that multi-disunion node algorithm is made to be performed for each extracted feature.

(3) Multi-Disunion Node

In order to detect such a multi-disunion node, an optional surface is selected first and detected by search of a disunion node, and another optional surface is eliminated together with the optional surface selected first and connection of the remaining surfaces is searched. As a result, if there are two subgraphs disconnected by the elimination of the two surfaces, these two surfaces are fixed upon as disunion nodes. After an attempt is made at forming the result matrix by adding in turn a surface to be eliminated in such a way, the disunion nodes are detected by retrieving if there are disconnected graphs. If the result matrix is divided into subgraphs, the above process is repeated even for the subgraphs. The multi-disunion node is obtainable by enlarging an incidence surface in this way and the above process is repeated until multi-disunion node algorithm is completely performed for all obtained basic shapes.

Therefore, features like a pocket, a clogged hole, a through hole, a clogged slot, a corner end, a slot and an end are automatically recognized and a process planing can be thereby executed.

(4) Process Plan Template

The present research unifies design and processing, makes it easy to work out a process planing automatically and shortens a period of product development by obtaining the data related to the process planing from three-dimensional geometric data in the process of designing and processing mechanical parts. The following Figure shows machining process template example for each atomic feature.

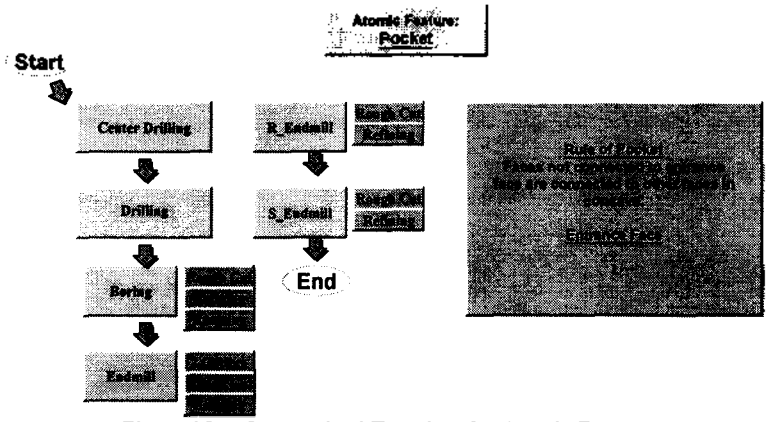


Figure 10. Summarized Template for Atomic Feature

Conclusion

Industrial enterprises are currently faced with fundamental changes in the overall application framework. Short innovation cycles, worldwide competition, and high costs are compelling companies to structure business operations rationally. The propsed component based development tool provides the component based programming environment for the development of distributed manufacturing applications to quickly, efficiently, and inexpensively develop all of the functions and modules desired by actual users in the industries. In this research, we applied the tool to the domain of the design and process plan, and propose a framework to retrieve existing similar design models in database, initiate a model, generate a process plan, and store the new model in the database automatically.

Reference

[1] Clemens Szyperski, Component Software : Beyond Object-Oriented Programming, Addison-Wesley, 1998

[2] Carma McClure, The Three Rs of Software Automation : Re-Engineering, Repository, Reusability, Prentice Hall, 1992.

[3] Jingwen Cheng, "A Reusability-Based Software Development Environment," ACM SIGSOFT, Vol. 19, No. 2, pp. 57-62, 1994.

[4] Eliseo Mambella, Roberto Ferrari et al, "An Integrated Approach to Software Reuse Practice," SSR '95, Seattle WA, USA, pp. 63-71, 1995.

[5] R Helm, I. Holland, and D. Gangopadhyay "Contracts: Specifying Behavioral Compositions in Object-Oriented Systems," Proceedings of OOPSLA/ECOOP, 1990.

[6] David Garlan and Mary Shaw, "An Introduction to Software Architecture," CMU Technical Report, 1994.

[7] Kazuo Matsumura, Akihiro Yamashiro, Toshiyuki Tanaka, and Ikumune Takahashi, "Modeling of Software Reusable Component Approach and its Case Study," IEEE, 1990.

[8] Hwa Gyoo Park, "Feature Recognition System Based on Neural Network Technique", Proceedings of 4World Congress on Expert Systems, 1998.

[9] Hwa Gyoo Park, "Expert Classification System for Concurrent Engineering", Proceedings of Concurrent Engineering '98 Japan, 1998.

[10] Orfali, Harkey, The Essential Client/Server Survival Guide, Wiley, 1996.

[11] Asrafi, n and Kuilboer, J., "The Information Repository : A Tool for Metadata Management", Journal of Database Management, 1995.

[12] Orfali, Harkey and Edwards, The Essential Distributed Object Survival Guide, Wiley, 1996.Hwa Gyoo Park, "Multi Agent System Based Manufacturing System Enhancement", Proceedings of Concurrent Engineering '98 Japan, 1998.