AGENT-BASED APPROACH FOR THE CONSTRUCTION OF A DESIGN SUPPORT SYSTEM FOR CONCEPTUAL CHEMICAL PROCESS SYNTHESIS

Chonghun Han* and George Stephanopoulos**

*Dept. of Chemical Eng., Automation Research Center, Pohang University of Science and Technology, Pohang, Kyungbuk, 790-784, Korea
(Tel: (0562) 279-2279; Fax: (0562) 279-2699; E-mail: chan@vision.postech.ac.kr)

**Dept. of Chemical Eng., Massachusetts Institute of Technology, Cambridge, MA 02139, U.S.A.
(Tel: (617) 253-3904; E-mail: geosteph@mit.edu)

Abstracts A successful, computer-aided design support system can help a process designer focus on making effective design decisions, not merely tedious routine calculations. Such a system is essential to enhance quality of design in terms of economics, environmental benignity, reliability, robustness, and operability. Such a statement is even more accepted when applied to conceptual design problems, where gross design specifications are given while a combinatorial number of design alternatives exists. This paper presents an agent-based approach as a systematic and efficient way to design a design support system for the synthesis of conceptual chemical processes. An agent-based approach allows us to handle design knowledge as an object and thus greatly improve the modularity and reusability of that knowledge. Such modularity and reusability lead to the increased productivity in the development of a design support system and the increased ease in the relaxation of design decisions and the generation of design alternatives, both of which functions are critically important in dealing with the complexity and uncertainty of conceptual design problems.

Keywords Agent; design support system; conceptual design; object model

1. INTRODUCTION

When chemists discover new reactions, the process designer has to determine if a process based on the discovery will be profitable, and if so, to determine the best few flowsheets. This design activity is called *conceptual design*. Only a small fraction of the total design effort (5 to 10 %) is spent on the conceptual design which, nevertheless, fixes about 85% of the total manufacturing cost, once the flowsheet has been fixed. However, compared to the limited information available at the conceptual design stage (desired product purity and flow rate, the set of reactions, etc.), the designer should decide the specification of the process units and their interconnections, including a separation system, an energy management system, reaction or separation system solvents, any waste treatment and disposal, and finally a set of attractive alternative flowsheets. Considering the complexity and uncertainty of conceptual design, the role of process designer as a decision-maker is critical for the successful design.

A design support system offers the role of an intelligent assistant to the desginer and allows the designer to focus on making effective decisions. As a rusult, the system can bring up enhancement of quality of design in terms of economics, reliability, robustness, and operability. The *means-end* procedure, used by Siirola and Rudd (1971) to develop the AIDES system, and BALTAZAR system of Mahalec and Motard (1977), which used *mechanical theorem-proving* techniques to synthesize process flowsheets are among the earliest efforts in systematizing the synthesis of conceptual design. This paper proposes an agent-based approach to model a synthesis process and develop a design support system. When a set of product specifications is given, the support system will generate a few good process flowsheets through the interaction with a process designer.

2. APPROACH

Requirements Analysis

The process of design is one of converting a set of design specifications into a set of physically realizable entities that can meet these specifications. The development of a design support system needs a model for such conversion process. Hierarchical approach in conceptual design (Douglas, 1988; Douglas and Stephanopoulos, 1994) has been the conceptual basis for the model. As the model describes how the physical description of the design must progress from a beginning state, which lacks details necessary to meet the specifications, to an end state where these details are present, the model needs the effective and expressive representations for the design states and the transitions.

Such representations are of two types: declarative and procedural. A modeling language, MODEL.LA. (Stephanopoulos et al., 1990a), has been selected as a language to represent declarative knowledge. This paper focuses on the representations of procedural knowledge of the design process and the design of a design support system. Spriggs (1990) has listed several requirements for the successful development of a design support system: it must be evolutionary, integrative, and with an ability to learn. We will discuss these issues in the following sections.

Principles-Based Task Decomposition. Task decomposition is a domain analysis methodology to overcome the complexity of large engineering tasks and identify the common tasks. A properly chosen decomposition allows high-level operations to be addressed somewhat independently from low-level details. Park (1994) reported a list of questions which guides the process of task decomposition. While such questions are very valuable, their *ad hoc* character limits the application of such guidelines to a broader range of problems.

Based on the *abstract refinement model* (Stephanopoulos, 1990), we introduce a *principles-based task decomposition* as a systematic way of decomposing a design process into a set of design tasks. The decomposition is performed based on the design principles (Han et al., 1995) listed below.

- Principle-1: Hierarchical planning of the design methodology.
- Principle-2: Successive refinement of specifications into implementations.
- Principle-3. Propagation of constraints.
  Propagation of constraint values
  Induction of new constraints.
- Principle-4. Hierarchical decomposition with coordination.
- Principle-5. Unified transformational design.
- Principle-6. Context-based design.

Fig. 1 shows as a domain model the generic goal structure developed for the design process of continuous, single product, vapor-liquid chemical processes. Note that the structure does not represent all the minor details of the design.

Task Refinement. Once a design process has been decomposed into tasks and the dependencies among the tasks identified, a number of analysis methods (Steward, 1981, Eppinger et al., 1991) can be applied to streamline the design process to minimize redundancy and identify the generic common properties shared by different tasks. These common properties lead to the creation of abstract classes which become the *agents*.

Representation of Design Tasks

Agent Definition. As a design agent models a design task, it has a design goal and a design plan about how to accomplish its goal. An agent can be defined as a tuple G=(I/O, KR, C), where:

- I/O is the Input/Output interface and defines the channels of information exchange with other agents;
- KR is the set of Knowledge Representation models required by the agent. It describes how knowledge should be structured to achieve the agent functionality.
- C is the Control structure which specifies the procedure for achieving the task's functionality by operating on the knowledge.

Agent Construction Through Mapping Domain Model To Agent Model. A domain model- a goal structure tree- is mapped to an agent model. The hierarchies and interdependencies among agents identified from the task decomposition are implemented. The structure of the agent model is designed from the mapping of the domain model, i. e. of the model representing the methodology for conceptual design, as defined by Douglas (1988) and Douglas and Stephanopoulos (1994).

Modeling Elements: Basic Agents. Basic agents are abstract classes (shown in Arial font), which provide common structure and behavior to a certain group of classes. They are the most fundamental building blocks for modeling a design process. More detail descriptions on each class are given in Han(1993).

1. **Generic Manager**: manage design agents.

1.1 **DesignManager**: refines a design from one abstraction layer to the more detailed one by managing the agents. Also manages a redesign process in case of design failures, i.e. design constraint violation or a poor economic performance.

1.2 **ProjectManager**: is in charge of a design project and organizes and coordinates the activities of design managers.

2. **DesignAgent**: a class with domain-specific knowledge on how to solve the given problem. The knowledge is organized as a design plan, which is used to resolve the conflicts occurring during the design. A design agent solves a given problem independently. When there is a conflict which the agent cannot resolve itself, it reports to its manager the conflict and returns the control.

2.1 **StructureAgent**: receives a boundary system from its manager and decomposes the system into a set of subsystems.

2.2 **CoordAgent**: finds the decomposed subsystems which do not have connections to other subsystems and establishes the connections among them.

2.3 **DrawAgent**: receives a list of **GenericUnits** from the manager, creates icons, decides their locations, and makes associations between the **GenericUnits** and the icons.

2.4 **ModelAgent**: is in charge of the analysis, in other words, sets up and solves the material and energy balances for the synthesized structure..

2.5 **EvalAgent**: is in charge of cost evaluation for the synthesized flowsheet.

Interface Elements: Basic Classes. The design support system integrates the human designer at every state of the evolving design by allowing the user to define the scope of design, guide the direction of the design evolution, or/and select among competing designs in a fast and efficient manner. A variety of interface classes such as **MainWindow, LayoutWindow, GraphicUnit, GraphicPort, GraphicStream,** are provided to help human designer interact with the system.

Semantic Relationships Among Design Agents. The following list of semantics establishes the requisite relationships among the various design-task elements described above. The resulting object model shown in Fig. 2 shows the key modeling elements and their relationships.

(1) "decompose": a**IOStructureAgent** decompose a**PlantSystem**.
(2) "is-decomposed-into", e. g. a**PlantSystem** is-decomposed-into (a**ReactionSystem**, a**SeparationSystem**).
(3) "transform", e.g. a**IODrawAgent** transform (a**PlantSystem**).
(4) "is-transformed-into", e.g.(a**ReactionSystem**) is-transformed-into (a**GraphicUnit**).
(5) "compute", e.g. a**RecycleModelAgent** compute a**RecycleFlowsheet**.
(6) "evaluate", e.g. a**RecycleEvalAgent** evaluate a**RecycleFlowsheet**.
(7) "refine", e.g. a**RecycleManager** refine a**PlantSystem** at the input/output structure.
(8) "is-refined-into", e.g. a**PlantSystem** at the input/Output structure level is-refined-into a**Flowsheet** at the recycle structure level.

System Design

Object Model. Modeling classes described in the previous sections are combined into a very powerful design support system as shown in Fig. 2. The conventions of object-modeling technique (Rumbaugh, 1991) were employed.

Note how objects are combined into a powerful abstraction and communicate with each other. The object model can be naturally used for the concurrent design as long as the design problem is correctly decomposed into subproblems and each subproblem is handled by each object. The object model can be implemented in any of the various object-oriented languages, such as C++, Smalltalk™, CLOS (currently, implemented in ACTOR™)

An Example. *ConceptDesigner*, a design support system for conceptual design of chemical processes, has been developed based on the agent approach. Fig. 3 shows the overall architecture of the *ConceptDesigner*. *ConceptDesigner* provides several facilities: (1) quick process design mainly by making design decisions, (2) integrated process analysis tool for material and energy balance, (3) cost analysis, (4) quick generation and screening of design alternatives, and (5) intelligent user interfaces. When a user has a correct design specification without any internal or logical conflict and makes design decisions quickly, he/she can generate the design within 15 to 20 minutes. Figure 4 shows a snapshot of the design process using *ConceptDesigner*.

3. DISCUSSION

The development of *ConceptDesigner* produced useful insights that are applicable to other design problems. These are considered in the remaining paragraphs.

Benefits of Agent-Based Approach. Agent-based approach provides a very flexible framework where a variety of design tasks can be smoothly integrated thanks to the encapsulation and well-defined interfaces of agents.. In comparison with conventional approaches, the advantages are twofold:

1. *Easy maintenance and adaptation*. The proposed architecture allows the problem to be solved by a network of agents. New agents can be easily added to the existing agents or an existing agent can be easily removed or modified. Furthermore, the autonomy of design agents enables the extension of the sequential design approach to the concurrent one in a very smooth manner through the reconfiguration of their networks, not the structures themselves.
2. *Easy generation of design alternatives*. The flexible architecture of agents enables us to generate many variations of design by changing the design decisions. Note that an agent works as an operator which detects the situation of the flowsheet, transforms the flowsheet into a different state.

Design Space. During a design process, three spaces exist: (1) A *goal space*, where a model of design process is represented as a hierarchy of design agents. Each agent has a specific design goal identified in the task decomposition step. The model does not change and works like a shell. (2) A *decision space*, where the design decisions of a user are dynamically stored as nodes of a binary tree. Thus, for each design project, a single decision tree is created and grows dynamically, as more and more decisions are made during a design process. (3) A *design space*, where the results of decisions are recorded and kept. A user can see flowsheets, models, the results of material and energy balances, and cost computations. Note also that the structures of decision space and a design space are unique for each design problem.

Interagent Communication. The effective communication of design information among design agents is one of key requirements for a successful design support system. Two different ways of communication exist among design agents; a direct one among a manager and an agent, and an indirect one through a flowsheet, which is an object-oriented blackboard representing an evolving design state. The latter one is common among design agents at the same level in the hierarchy and has been commonly used in blackboard system research. When an agent makes a change in the blackboard, this change is monitored and responded appropriately by all the other agents which work on the blackboard.

Human Interaction. Human interaction is essential for two reasons: (1) It is impossible to capture and encode into a computer program all of the case-specific rules and constraints that may apply to a particular design problem. (2) Computer programs cannot handle NP-complete problems efficiently without human guidance. Thus, it is essential that the designer be given freedom to apply his/her creative and intuitive capacity to expedite the design process, including the power to modify and override solutions suggested by the system.

4. CONCLUSIONS

This paper presented a brief overview of an agent-based approach to develop a design support system for conceptual chemical process synthesis. We showed that the flexibility and concurrent character of this approach is very promising as a way of designing and implementing a successful design support system. Furthermore, this approach is a very natural extension to the paradigm of agents of the current modeling language available in chemical engineering area.

ACKNOWLEDGEMENT

The first author would like to acknowledge the Korean Science and Engineering Foundation and the Automation Research Center in POSTECH for financial support.

REFERENCES

[1] Douglas, J. M., *Conceptual Design of Chemical Processes*, McGraw-Hill., 1988.

[2] Douglas, J. M, and G. Stephanopoulos, "Hierarchical Approaches in Conceptual Process De Framework and Computer-Aided implementation," in *Proc. Foundations of Computer-Aided Process Design*, 1994.

[3] Eppinger, S. D., Whitney, D. E., Smith, R. P., and Gebala, D. A., "Organizing the tasks in complex design projects," In *Computer-Aided Cooperative Product Development* (Sriram, D. and Logcher, R., Eds.), pp. 229 - 252, Springer-Verlag, New York, 1991.

[4] Han, C., Human-aided. *Computer-based Design Paradigm., 2e Automation of Conceptual Process Design*, Ph. D. Thesis, Department of Chemical Engineering, Massachusetts institute of Technology, Cambridge, Massachusetts, 1993.

[5] Han, C., J. Douglas, and G. Stephanopoulos, "Automation in Design: The Conceptual Synthesis of Chemical Processing

Schemes," in *Intelligent Systems in Process Engineering* (G. Stephanopoulos and C. Han, Eds.), Academic Press, 1994.

[6] Mahalec, V. and R. L. Motard, " Procedures for the Initial Design of Chemical Processing Systems," *Comput. Chem. Engng*, l, pp. 57, 1977.

[7] Park, H,, M. R, Cutkosky, A. B. Conru, and S. Lee, "An agent-based approach to concurrent cable harness design," *Artificial Intelligence for Engineering Design, Analysis, and Manufacturing*, pp. 45 - 61, 1994

[8] Rumbaugh, I. M., M. Blaha, W. Premerlani, F. Eddy, and W Lorensen, *Object-Oriented Modeling and Design*, Prentice Hall, Englewood Cliffs, 1991.

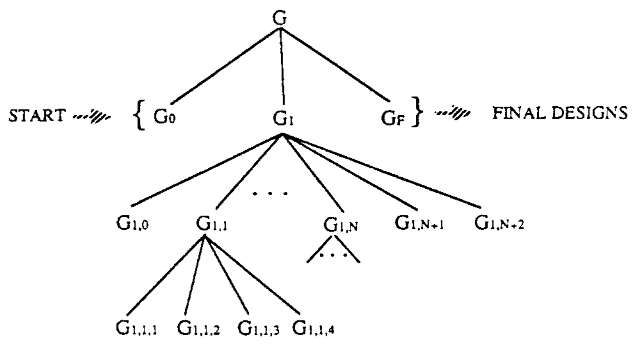
[9] Siirola, J. J. and D. F. Rudd, "Computer-Aided Synthesis of Chemical Process Designs," *Ind. & Eng. Chem.*, **19**, pp. 353, 1971.

[10] Spriggs, H. D. "Introduction: Additional Chemical Process Design issues," in *Foundations of Comp Aided Process Design* (J.J. Siirola, I. E. Grossmann and G. Stephanopoulos, Eds.), pp. 209 - 21 1, Elsevier, 1990.

[11] Stephanopoulos, G., "Artificial intelligence and Symbolic Computing," in *Foundations of Computer-Aided Process Design* (J.J. Silrola, I. E. Grossmann and G. Stephanopoulos, Eds.), Elsevier, 1990.

[12] Stephanopoulos, G., G. Henning and H. Leone, "MODEL. LA.: A Modeling Engineering. Part I: The Formal Framework," *Comput. chem. Engng.* pp. 813~846, 1990a.

[13] Steward, D. V. "The design structure system: A method for managing the design of complex systems," *IEEE Trans. Eng. Mgmt.* EM-28, pp. 71-74, 1981.



G: Design the conceptual processing scheme for a multi-step reaction plant
G_0: Initialize the project's specifications
G_1: Synthesize the process flowsheet for the whole plant designs
G_F: Evaluate process flowsheet and create imporved designs
$G_{1,0}$: Synthesize the plant complex structure

Figure 1. Tree of goal structure in *abstract refinement*

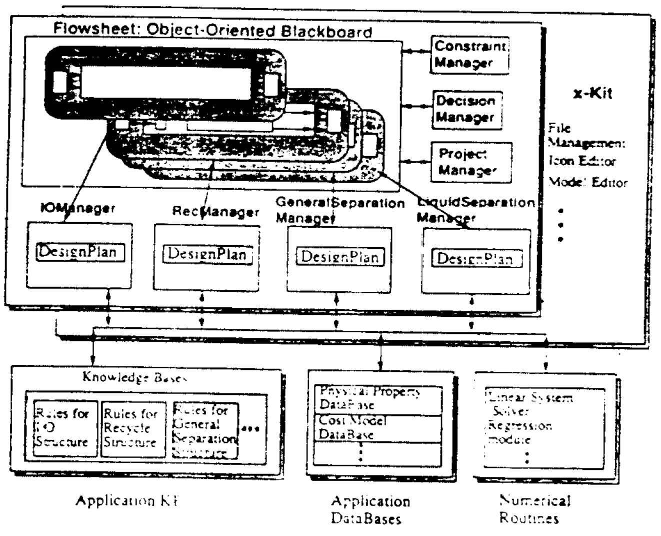


Figure 3. Overall architecthre of *ConceptDesigner*

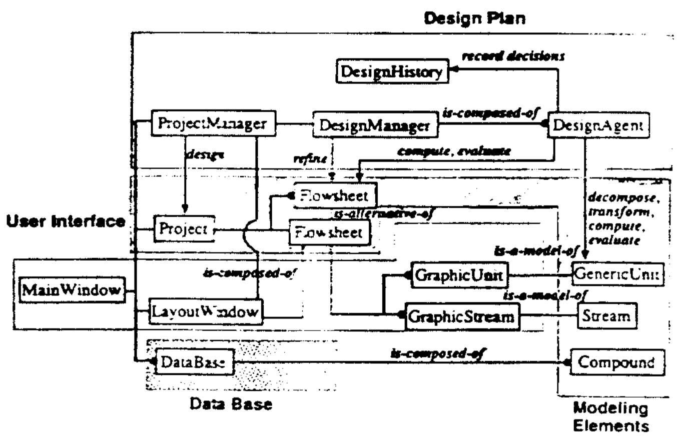
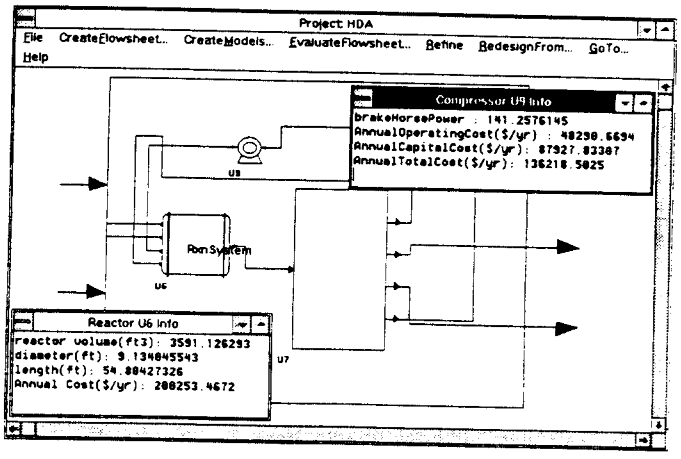


Figure 2. Object model for *ConceptDesigner*



```
Compressor U9 Info
brakeHorsePower : 141.2576145
AnnualOperatingCost($/yr) : 48290.6694
AnnualCapitalCost($/yr): 87927.83387
AnnualTotalCost($/yr): 136218.5025
```

```
Reactor U6 Info
reactor volume(ft3): 3591.126293
diameter(ft): 9.134845543
length(ft): 54.88427326
Annual Cost($/yr): 288253.4672
```

Figure 4. A snapshot of the design process using *ConceptDesigner*