Proceedings of IDETC/CIE 2006
ASME 2006 International Design Engineering Technical Conferences
& Computers and Information in Engineering Conference
September 10-13, 2006, Philadelphia, Pennsylvania, USA

DETC2006-99286

A MULTI-AGENT SYSTEM FOR MODULAR PLATFORM DESIGN IN A DYNAMIC ELECTRONIC MARKET ENVIRONMENT

Seung Ki Moon[1], Soundar R.T. Kumara[2], and Timothy W. Simpson[3*]
Department of Industrial & Manufacturing Engineering
The Pennsylvania State University,
University Park, PA 16802 USA

ABSTRACT

Electronic markets and web-based supply chain management have improved traditional product development processes by increasing the participation of customers and applying various trading processes between companies and suppliers in a dynamic electronic market environment. A multi-agent system is an appropriate tool to implement a product development system in a distributed environment because of its flexibility, scalability, and adaptability. This paper introduces a multi-agent system (MAS) based on market mechanisms to support modular platform design. The agent architecture for the proposed MAS is described, including specific agent roles, knowledge, and strategies. In particular, a reputation mechanism is used to select stable and reputable modules for the platform by detecting and dismissing volatile modules in a dynamic electronic market environment. We demonstrate the implementation of the proposed MAS using a multi-agent development framework and how to use module reputation for selecting a module for the platform. Through experiments, we illustrate that the MAS can be used to design modules in a product platform using the proposed market mechanisms.

Keywords: Multi-agent system, Product family design, Product platform, Market mechanisms

1. INTRODUCTION

Electronic markets and web-based supply chain management have improved traditional product development processes by increasing the participation of customers and applying various trading processes between companies and suppliers in a dynamic electronic market environment. With the potential of reducing transaction costs, the applications of electronic markets are dramatically increasing in various industries [1]. The growing number of electronic markets in product development has significantly increased information related to design and the complexity of transactions, making it difficult to control electronic markets with human resources [2]. Agent-based technologies provide a natural means to achieve information integration in such a distributed environment [3].

In today's competitive market, companies are increasing their efforts to reduce costs and time when developing new products while satisfying individual customer needs [4]. Companies also seek to maximize resource utilization by sharing and reusing distributed design knowledge and information when developing these new products. Product family planning is a way to achieve mass customization by allowing highly differentiated products from a platform while targeting individual products to distinct market segments [5].

Most previous work related to product design and multi-agent systems has been focused on the agents' roles and tasks in determining a suitable manufacturing environment. In product family design, a method to produce a variety of products should be considered for both dynamic and various market segments. A multi-agent system can help achieve higher levels of flexibility, scalability, and adaptability in a dynamic and distributed environment [6]. A multi-agent approach can be applied to develop an appropriate method to model a market-based product design system for the following reasons: (1) multi-agent systems and market-based mechanisms are inherently distributed, (2) products can be designed with modules whose information is distributed across a market environment, and (3) each module can be modeled as a self-interested agent – an autonomous decision-maker and a specific information holder at the same time.

[1] Graduate Research Assistant.
[2] Distinguished Professor of Industrial & Manufacturing Engineering and Member ASME.
[3*] Professor of Mechanical and Industrial Engineering and Member ASME. **Corresponding Author**. Email:tws8@psu.edu. Phone/fax:(814) 863-7136/4745.

In this paper, we introduce a multi-agent system (MAS) based on market mechanisms to support module-based product family design. The architectures of agents in the MAS are described including specifying their roles, knowledge, and strategies. A reputation mechanism is applied to select a reputable module that is less influenced by market fluctuation. The objective in this paper is to identify and configure a module-based platform using a reputation mechanism in a dynamic electronic market environment. The reputation mechanism focuses on selecting reputable modules for the platform by detecting and dismissing volatile modules. In order to take advantage of having a product family, a module involved in a platform should be more stable in an electronic market than the other modules that are unique to individual products.

This paper is organized as follows. In Section 2, a literature review for a multi-agent system and product family are discussed. Section 3 introduces module-based platform design for a product family and electronic markets. Section 4 presents the architectures of agents in the MAS based on a reputation mechanism. In Section 5, the MAS is implemented, and an experiment and analysis of selecting modules are described. Closing remarks and future work are presented in Section 6.

2. LITERATURE REVIEW

As companies strive to minimize cost and time for developing new products by sharing and reusing distributed design knowledge and information, multi-agent systems provide an ideal mechanism to efficiently develop various products by integrating distributed design knowledge and information [7]. Madhusudan [8] developed a flexible agent-based coordination framework for new product development in a distributed design process system. Jia, et al. [9] presented an agent-based system for coordinated product development and manufacturing that is able to execute all the tasks in a coordinated and flexible way. Anumba, et al. [10] introduced a multi-agent system framework to facilitate collaborative design and interaction protocols for agent negotiation and applied it to designing industrial buildings. Tan, et al. [11] developed a multi-agent framework to provide information that helps designers, engineers, and managers work together to improve initial designs by satisfying a wider variety of concerns.

In agent-based electronic markets, reputation is often used to detect and dismiss fraudulent agents [2, 12]. Zacharia, et al. [13] presented a framework for agent-mediated knowledge marketplaces in which agents' reputations are established by dynamic pricing algorithms. Padovan, et al. [2] described the prototypical implementation of an automated subsequent treatment of reputation information in a multi-agent system. Tran and Cohen [12] proposed a reinforcement learning and reputation algorithm-based algorithm for buyers and sellers in agent-based electronic marketplaces that maximized expected value of goods for buyers and expected profit for sellers.

A product family is a group of related products based on a product platform, facilitating mass customization by providing a variety of products for different market segments cost-effectively [14]. A successful product family depends balancing the trade-offs between the economic benefits and performance losses incurred from having a platform. Simpson, et al. [15] categorized platform design approaches as either top-down (proactive platform) or bottom-up (reactive redesign). The top-down approach manages and develops a group of products based on a product platform, while the bottom-up approach seeks to redesign an existing set of products around a platform. Moore, et al. [16] used conjoint analysis to determine a product platform. Siddique and Rosen [17] described a method to design a platform from an existing group of products by comparing commonalities in the assembly process. Gonzalez-Zugasti, et al. [18] designed platform modules to minimize design risk and save costs relating to developing a product family. Simpson, et al. [15] introduced a method to optimize a platform by minimizing performance loss and maximizing commonality. Rai and Allada [19] introduced a two-step approach to determine a modular platform for a product family, which consists of an agent-based optimal technique and post-optimization analysis using the quality loss function.

3. MODULAR PLATFORM DESIGN AND ELECTRONIC MARKETS

3.1 Module-Based Product Families

The basic idea of modular design is to organize products as a set of distinct components that can be designed independently and develop a variety of products through the combination and standardization of components [20]. Modules are achieved by decomposing the functions of products into independent sub-functions in which interaction or interdependence between sub-functions is minimized [21]. The modules make it easier to reuse in different products, allowing development and manufacturing costs to be significantly reduced [22].

The modules can be categorized based on function into unique modules and common modules. Unique modules are based on distinctive functions within a product family and can be not replaced by those in the different module to fulfill their task. Common modules are based on common functions within a product family and can be shared. In terms of a dynamic electronic market, modules can be separated into two types based on their design: 1) unique design modules and 2) alternative design modules. Unique design modules are designed by a unique design method or are provided by one supplier. Alternative design modules can be developed by various design methods or provided by several suppliers. Figure 1 shows the relationship between modules in a product family.

A well-defined platform reduces production costs by improving economies of scale and reducing the number of different components that are assembled. Suppose that a product family consists of unique modules and common modules as illustrated in Figure 1. A platform is defined as the set of common modules.

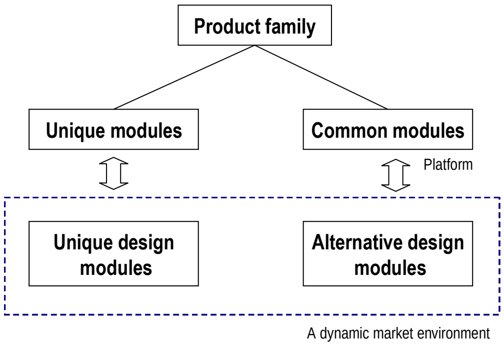


Figure 1: Relationship among Modules in a Product Family

As shown in Figure 2, we propose the process of developing a product family based on customer needs (CNs). In the initial phase, CNs are analyzed to understand customer intent and determine a strategy for developing a product family. For example, the number of products is decided by customer groups that are classified according to CNs. CNs are also used to identify appropriate functional requirements (FRs) and then mapped to them. FRs describe a product's behavior and features that are defined by technical information and data for its design. Products are developed based on FRs, and their functional modules are determined. For the product family, finally, a platform is identified that consists of several common modules.

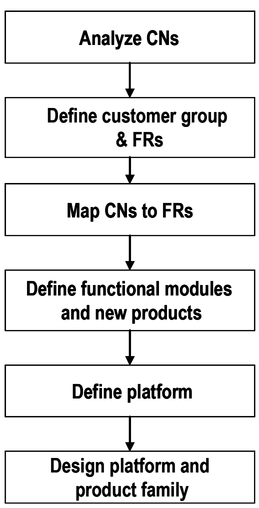


Figure 2: A Process for Developing a Product Family

3.2 Electronic Market

A dynamic environment follows rudimentary electronic market (e-market) features such as business behaviors between buyers and sellers, dynamic pricing, adjusting attributes, and alternative selections [2, 12]. This e-market provides an agent environment where agents are economically motivated. The nature of an e-market allows economic agents (buyers and sellers) to freely enter or leave the e-market and negotiate with each other to obtain economic benefit. As shown in Figure 3, there are two types of agents for module-based product design in an e-market: buyers and sellers. Buyers are defined as auctioneers and sellers as bidders, and their goal is to maximize their own benefit. The role of a buyer is to purchase a module designed by a seller. Sellers can provide alternative modules depending on their strategy and market conditions. Buyers can access all relevant sellers by querying information from them.

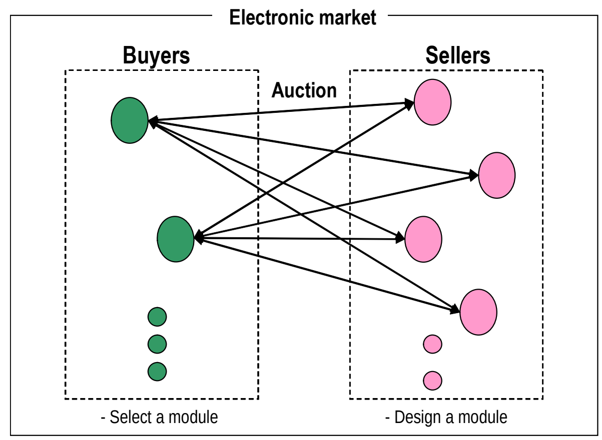


Figure 3: Agents for Module Design in Electronic Market

In this paper, module reputation is defined as module stability in a dynamic e-market. Module stability is represented by the degree of the variation of module design in the market and can be affected by a company's market strategy, design technology and trends, component quality, and production cost. A stable and reputable module is less affected by market fluctuation. In a dynamic e-market, a module can be designed by alternative design methods. The cost and quality of the module may have different values depending on the design methods or the components used. To choose stable and reputable modules effectively, we propose a multi-agent system based on such a market mechanism. The next section introduces the proposed multi-agent system in detail.

4. A MULTI-AGENT SYSTEM FOR MODULE DESIGN

4.1 Developing the Multi-Agent System

Software agents provide an ideal mechanism to realize information integration in design and manufacturing. An agent has access to at least one and potentially many information sources and is able to collate and manipulate information obtained from these sources in order to answer queries posed by users and other information agents [23]. Agents have been used extensively in product design and can be used in product family design if developed properly.

An agent-based technique based on agents' roles and tasks can provide appropriate methods to solve product design problems [7, 19, 23]. To facilitate the process of developing a module-based product family, a multi-agent system (MAS) is developed based on an electronic market environment. As seen in Figure 4, there are four types of agents in the proposed multi-agent system: 1) a coordinator agent (CA), 2) a platform agent (PA), 3) module agents (MAs), and 4) design agents (DAs).

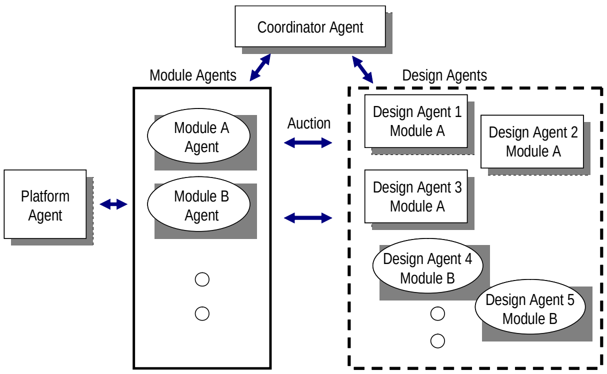


Figure 4: Multi-Agent System for Modular Platform Design

The main task in the proposed MAS is to determine platform modules by selecting appropriate common modules and subtasks for design modules. The PA decomposes the main task into subtasks based on the product's functions and assigns the module design (subtasks) to MAs by matching MAs' roles and tasks. The CA manages the coming and leaving agents, the MAs' requirements, and the DAs' design items. Based on module reputation, MAs fulfill the requested tasks with DAs using an auction and return the result to the PA. After MAs perform their tasks, the information of the module reputation is translated into new knowledge for sharing and reusing. The number of MAs is determined by the number of subtasks generated. A subtask is defined as designing a module to satisfy its functional requirements. Within a dynamic e-market, a module can be designed from a variety of components fulfilling the same function or from various suppliers. DAs can provide alternative modules in terms of cost and quality according to DAs' strategy or market situation.

Table 1: Agents' Roles and Knowledge for Modular Platform Design

Agent	Roles	Knowledge
Coordinator Agent (CA)	• System management • Track the coming and leaving agents • Record information from MAs and DAs • Connect MAs and DAs	• MAs and DAs information
Platform Agent (PA)	• Decompose tasks • Make decisions: select module(s) for platform • Allocate tasks (resource)	• Product design • Decomposition algorithm • MAs information
Module Agent (MA)	• Make decisions: select module • Request to design agents • Evaluate module quality • Update module reputation	• Strategy for negotiation • Learning algorithm • DAs reputation • CA information • Module information
Design Agent (DA)	• Design a module • Search components	• Module and component information • Searching algorithm • Strategy for negotiation • CA information

In the proposed MAS, agents use knowledge to decide actions for performing their roles. The knowledge can consist of constraints, functions, rules, and facts, which are associated with product design and the system environment. Since agent activities are determined by knowledge, knowledge must be related to the overall system tasks and be accessible in an appropriate form [24]. Knowledge for representing products, modules, and components can be represented by the Techspecs Concept Ontology (TCO) [25]. Knowledge defined using TCO can help represent information for agents in a collaborative and distributed environment. Knowledge related to module design development is stored in a knowledge base and used to define agents' activities and tasks. The roles and knowledge of each agent are listed and described in Table 1. In this paper, reasoning about knowledge is used for inference and to capture knowledge in a distributed environment.

4.2 Interaction and Communication Design

Software agents need to be able to interact and communicate with other agents in order to cooperate and to share knowledge. Therefore, a common or inter-translatable representation language and a framework of knowledge to interpret the exchanged messages are necessary. In this paper, protocols for communication between agents are based on the Foundation for Intelligent Physical Agents (FIPA) [23]. FIPA defines a common format for messages. Each message has a performative and a number of parameters (attributes may be thought of as information for tasks). Figure 5 shows the sequence diagram of agent communication language (ACL) interactions in MAS.

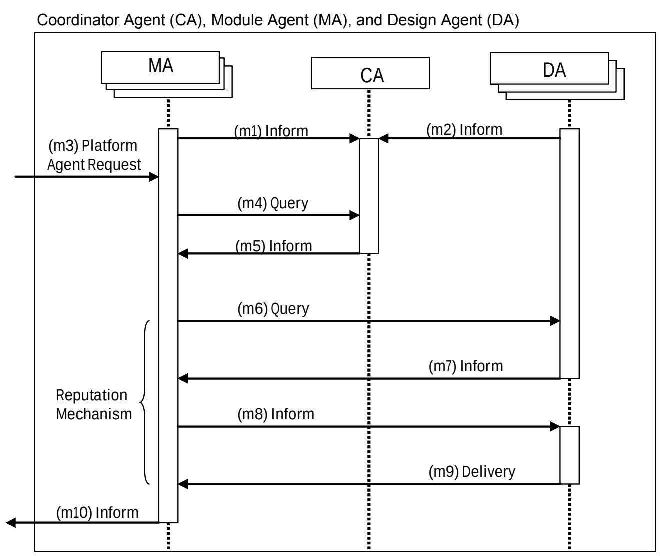


Figure 5: ACL Interactions (Sequence Diagram)

ACL messages and attributes in the FIPA protocol are summarized in Table 2 and explained as follows.

- Interaction *m1-m2-m3-m4-m5*: In MAS, if MAs and DAs are entering into the market, they send their information to the CA using an inform message *m1* and *m2*. The CA records the MAs' and DAs' information. The PA sends a request message *m3* to the MAs, sending new module specifications and information to design a platform. Based on the message *m3*, the MAs send a query message *m4* to the CA to obtain the DAs' information. The CA sends the DAs' information to the MAs using message *m5*.

- Interaction *m6-m7-m8-m9*: The MA sends a query message *m6* to the DAs, sending new module specifications and information to request a design. The DAs who received the request prepare their bid and enter the auction. To enter the auction, the DAs send inform message *m7* to the MA. After collecting the deals from the DAs for designing a module, the MA determines a winning DA based on the DAs' reputations. The winning information is sent to the DA via an inform message *m8*. The DA sends the module to the MA via a delivery message *m9*. The MA evaluates the delivered module and updates the DA's reputation.

4.3 Decision-Making and Learning Algorithm

Module reputation can be affected by a company's market strategy, design technology and trends, component quality, and production cost. In this paper, we consider the quality and price of modules as the reputation factors related to market fluctuation. To develop a learning and reputation-based algorithm for the module agent (MA) and design agent (DA), the approach of Tran and Cohen [12] is applied as follows.

In the proposed MAS, suppose that a MA requests some modules to determine a good module for a platform. Let M be the set of modules, P be the set of price, and I be the set of all MAs, and D be the set of all DAs in the marketplace. M, P, I, and D are finite sets. A MA determines the reputation of all DAs in the market using function r^{MA}: $D \mapsto (-1, 1)$, which is called the *reputation function* of the MA. The reputation can be categorized based on the value of the function: (i) reputable ($r^{MA} \geq \Theta$, $0 < \Theta < 1$), (ii) disreputable ($r^{MA} \leq \theta$, $-1 < \theta < 0$), and (iii) non-reputable ($\theta < r^{MA} < \Theta$), where Θ is a *reputation threshold* and θ is a *disreputation threshold*. Non-reputable means that a MA does not determine the reputation of a DA because of insufficient information. A reputation value is set to 0 initially and updated depending based on the transaction.

Let D_r^{MA} and D_{dr}^{MA} be the set of reputable and disreputable DAs to a MA, respectively. To select a good module and update the reputation of a DA, a MA uses a utility function. The utility function (u_i) of a module for a MA_i is calculated by the difference between the expected module value (f_i) and the true module value (v_i):

$$u_i = v_i - f_i \tag{1}$$

where f_i is estimated by an expected module value function $f_i : M \times P \times D \mapsto \Re$. The real number $f_i(m,p,d)$ represents the MA's expected module value of designing module m from DA_d paying price p. Meanwhile, v_i is determined by examining the quality of the module provided from the DA_d and estimated by a true module value function $v_i : M \times P \times Q \mapsto \Re$, where Q is a finite set of real values representing module quality.

Since DAs may offer the module m with different qualities and a DA may alter the quality of its modules based on its market strategy, the MA gives more trust to DAs with a good reputation and chooses a DA with maximum expected module value among the reputable DAs in D_r^{MA}. If there are no DAs in D_r^{MA}, then MA randomly chooses a DA with a small probability ρ in the set of non-reputable DAs. The utility value is used for learning the expected module value function by a reinforcement learning mechanism:

$$f_i(m, p, d) \leftarrow f_i(m, p, d) + \alpha u_i \tag{2}$$

Table 2: ACL Message and Attributes in Protocol

Msg No.	Sender	Receiver	Performative	Content Description	Attributes (data type)
m1	MA	CA	Inform	MA information	1. MA ID, IP Address (Integral)
m2	DA	CA	Inform	DA information	1. DA ID, IP Address (Integral) 2. DA's design item (Sting)
m3	PA	MA	Request/ Send	A new platform design Module specifications	1. Module information – functions (String) 2. Module specification - attributes (size, weight, quality, technical specs.) – (String, floating) 3. Quantity (Integral)
m4	MA	CA	Query	Module information	1. Module information – functions (String)
m5	CA	MA	Inform	DA information	1. DA ID, IP Address (Integral) 2. DA's design item (Sting)
m6	MA	DA	Query	Module specifications	1. Module specification – functions, attributes (cost, weight, technical specs, assembly specs.) - (String, floating) 2. Quantity (Integral)
m7	DA	MA	Inform	Module information	1. Module information – cost, weight, technical specs, assembly specs (String, floating)
m8	MA	DA	Inform	Acceptance	1. Message for acceptance (String)
m9	DA	MA	Delivery	Modules	1. Selected Modules
m10	MA	PA	Inform	A new platform design	1. A new platform information

where α is the learning coefficient ($0 \leq \alpha \leq 1$). If $u_i \geq 0$, then the expected module value is updated with the same or a greater value than before. In this case, a chance to choose DA_d is increased if DA_d provides a continuously good module m at price p in the next auction. Otherwise, if $u_i < 0$, then the expected module value is updated with a smaller value.

The reputation rating r^{MA} of a DA needs to be updated according to updating the expected module value. Let $p_i(m) \in \Re$ be the module value that a MA demands for the module m. Based on an approach proposed by Yn and Singh [26], the following reputation updating calculation is used [12]:

If $v_i(m,p,q) - p_i(m) \geq 0$, then the reputation rating $r^{MA}(d)$ is increased by:

$$r^{MA}(d) \leftarrow \begin{cases} r^{MA}(d) + \beta(1 - r^{MA}(d)) & \text{if } r^{MA}(d) \geq 0 \\ r^{MA}(d) + \beta(1 + r^{MA}(d)) & \text{if } r^{MA}(d) < 0 \end{cases} \tag{3}$$

where β is a positive factor called the cooperation factor (β>0) that is defined as:

$$\beta = \begin{cases} \dfrac{v_i(m,p,q) - p_i(m)}{\Delta v_i} & \text{if } \dfrac{v_i(m,p,q) - v_i(m)}{\Delta v_i} > v_{\min} \\ \beta_{\min} & \text{otherwise} \end{cases} \tag{4}$$

where $\Delta v_i = v_{i,\max} - v_{i,\min}$ with $v_{i,\max}$ and $v_{i,\min}$ being the maximum and minimum value of the true module function. If $v_i(m,p,q) = p_i(m)$, then the value $\beta_{\min}$ is used to prevent β from becoming zero.

If $v_i(m,p,q) - p_i(m) < 0$, then the reputation rating $r^{MA}(d)$ is decreased by:

$$r^{MA}(d) \leftarrow \begin{cases} r^{MA}(d) + \gamma(1 - r^{MA}(d)) & \text{if } r^{MA}(d) \geq 0 \\ r^{MA}(d) + \gamma(1 + r^{MA}(d)) & \text{if } r^{MA}(d) < 0 \end{cases} \tag{5}$$

where γ is a negative factor called the non-cooperation factor ($\gamma < 0$), which is defined as:

$$\gamma = \lambda\left(\frac{v_i(m,p,q) - p_i(m)}{\Delta v_i}\right), \tag{6}$$

where λ is a penalty factor ($\lambda > 1$). To ensure that a reputation is difficult to increase and easy to decrease, $|\lambda|$ should be greater than $|\beta|$. According to the result of updating the reputation rate, a DA is reallocated to the new set of the reputation with a new reputation rating.

DAs' decision-making and learning algorithms are used to update their module price and quality to reflect the result of the transactions. DAs estimate their expected profit using an expected profit function, $k_d : M \times P \times I \mapsto \Re$. The real number $k_d(m,p,i)$ represents the DA's expected profit when designing module m, if MA_i selects the module m with price p. Let $c_d(m,i)$ be the cost of DA_d to design module m for MA_i. DAs choose a price greater than or equal to the cost of designing module to maximize their expected profit. The expected profit function is learned by a reinforcement learning mechanism:

$$k_d(m,p,i) \leftarrow k_d(m,p,i) + \alpha(t_d(m,p,i) - k_d(m,p,i)) \tag{7}$$

where $t_d(m,p,i)$ is the true profit of the DAs and is defined as follows [12]:

$$t_d(m,p,i) = \begin{cases} p - c_d(m,i) & \text{if DA is determined as the member of a platform} \\ 0 & \text{otherwise} \end{cases} \tag{8}$$

5. IMPLEMENTATION AND EXPERIMENTATION

To demonstrate the proposed MAS, we implemented the framework using JADE[4] (Java Agent Development framework) and JARE[5] (Java Automated Reasoning Engine). JADE is a software framework to develop agent applications that use FIPA specifications to manage agent communication. JARE is an environment for doing logical inference in Java. JARE can be used to model an agent's knowledge base. The objective is to determine appropriate platform module(s) for a product family using the module reputation mechanism subject to a dynamic e-market environment. The implementation focuses on negotiation between a MA and DAs to select a module.

5.1 Scenario and Agent Development

For implementing and evaluating the proposed MAS, consider a scenario where a platform consists of two different design modules. Each module can be designed using four different design methods that affect its quality. A module is selected by the module's design strategy for the platform. Based on this scenario, we consider an e-market populated with one coordinator agent (CA), two module agents (MAs), and four design agents (DAs). Since JADE is a type of middleware and a framework to develop multi-agent systems, we can use JADE's capabilities to perform the functions of a CA instead of developing the CA separately. Two MAs are developed and have two different strategies to choose an appropriate module. In this scenario, a module's price, cost, and quality are considered as reputation factors for determining a module's reputation value in the dynamic e-market. The cost and price of each module depends on its quality. In order to compare alternative module designs from different DAs, four DAs are developed that have alternative design strategies. Through experimentation, we expect that two MAs are trying to select stable and reputable modules based on the platform's strategy in the dynamic e-market.

To inference and capture the information for selecting modules, the knowledge of each agent is developed based on the agents' roles. Figure 6 shows examples of rules, stationary facts, and dynamic facts in the knowledge base for the MAs.

[4] http://jade.tilab.com
[5] http://jare.sourceforge.net

Rules:

((is_reputable ?DA)

(has_reputation ?DA ?value)

(>= ?value))

((is_disreputable ?agent) Θ

(has_reputation ?DA ?value)

(<= ?value))

((is_nonreputable ?DA)

(has_reputation ?DA ?value)

(< ?value)

(> ?value))

Stationary facts:

((self ?name_of_myself)) Θ θ

Dynamic facts:

((has_expected_value ?module ?price ?DA ?value))

((has_reputation ?DA ?reputation))

((has_trades_with ?DA ?number))

((has_average_quality ?DA ?quality))

((has_average_price ?DA ?price))

Figure 6: The Knowledge Base for MAs

5.2 Preliminary Experiment and Analysis

Based on the aforementioned scenario, six agents were developed as MAs and DAs for the experiment of selecting modules for a platform in a dynamic e-market, as shown in Figure 7. In the experiment, two MAs purchased the same module 100 times from four DAs and learned from the transaction history. Each experiment was performed 20 times to compare and analyze the behavior of the two MAs. We used finite and discrete values for the price, which varied randomly from 100 to 2000. The quality is represented by the cost of the module. We assume that the module quality has a normal distribution with mean 1000 based on the cost range. The MAs' strategies and the DAs' alternative design strategies are:

- MA1 uses ($2\times$ *quality – price)* as the module value function, i.e., module quality is twice as important as module price.
- MA2 uses (*price – quality)* as the module value function, i.e., module price and quality are equally important.
- DA1: adjusts modules' quality based on request and initial quality is 1000.
- DA2: provides a module with a fixed average quality value (q=1000).
- DA3: provides a module with quality chosen randomly from the interval [100, 2000].
- DA4: first tries to attract a MA with high quality (q=1500) and then cheats them with very low quality (q=300).

Parameters related to the learning and reputation algorithm are defined as follows:

- The threshold value for a reputable DA is 0.3 and a disreputable one is -0.3.
- The learning rate α and exploration rate ρ are both 0.9999 and decrease until they reach 0.1.
- The penalty factor λ is 1.5, which makes building reputation 50% harder than deconstructing the reputation.

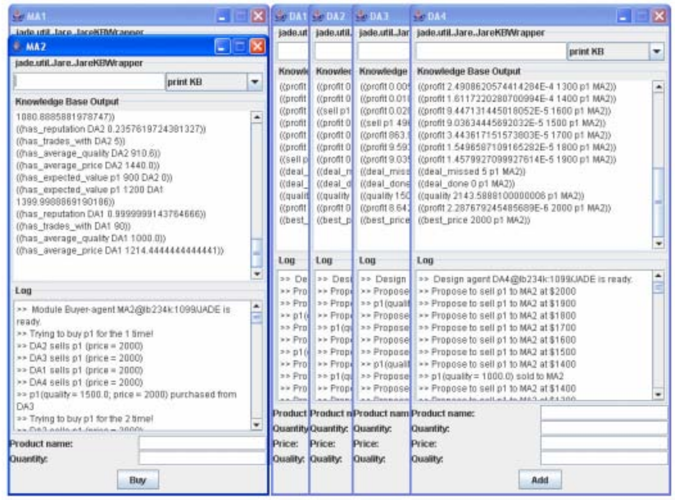


Figure 7: Screenshot of GUI for MAs and DAs

Figure 8 shows the number of purchases between DAs with different strategies. MA1 selected more modules from DA1 and DA2 than DA3 and DA4, since the average module quality of DA1 and DA2 is higher than the others. MA2 also preferred to purchase modules from DA1 and DA2. These results can be interpreted that the MAs selected reputable modules by detecting and dismissing disreputable modules for the platform that are less affected by market fluctuation.

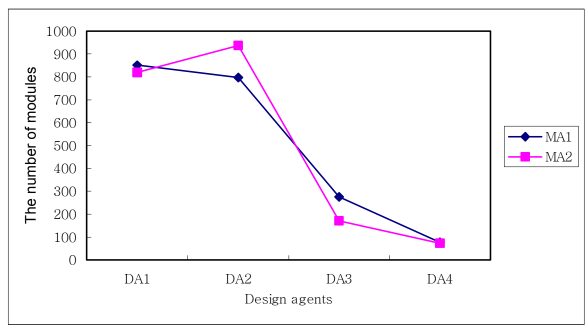


Figure 8: Number of Modules Selected by MAs

As shown in Figure 9, the random strategy of DA3 worsened its reputation. DA4 should have the worst average reputation; so, the number of purchases from DA4 is the lowest. Figure 9 shows the average final reputation values for the different DAs for MAs. As we expected, DA4 had a very low reputation value < –0.3, making DA3 a non-reputable agent. DA2 had a higher reputation value than DA1, but for MAs, both of them were good enough to be considered equally as reputable. As a result, MA1 selects DA1's design module for the platform design because MA1's strategy focuses on the quality of its module. Since MA2's strategy considers module's quality and price simultaneously, DA2's design module can be considered as the module of MA2 for the platform.

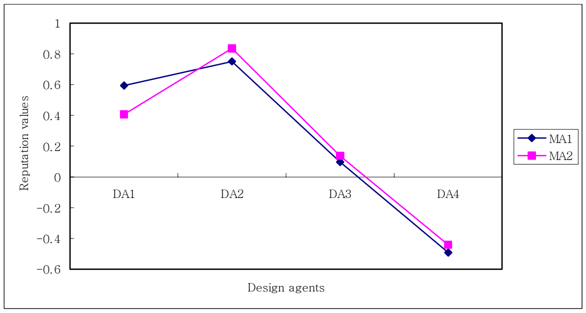
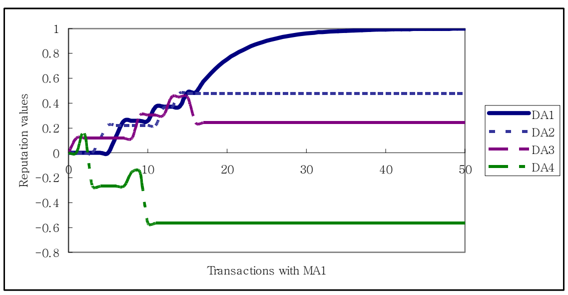
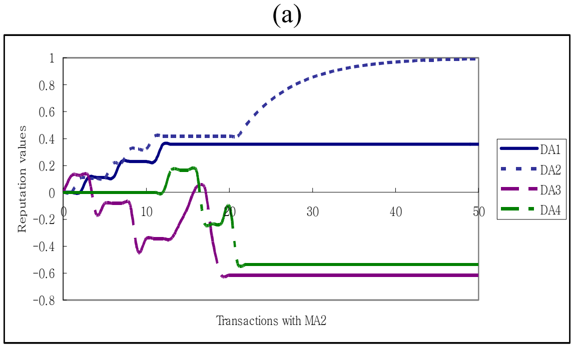


Figure 9: Average Final Reputation Values of DAs by MAs

Figure 10 illustrates the comparison of DAs' reputation values based on the DAs' strategies. Initially, MAs try all DAs for selecting modules. According to their module reputation values determined by the results of transactions, MAs select reputable modules to satisfy their strategies.



(a)



(b)

Figure 10: DA Reputation Values for MA1 (a) and MA2 (b)

We performed one-way Analysis of Variance (ANOVA) to determine whether any significant differences existed between selecting modules with different strategies based on the experimental results. In this test, the level of significant (p-value) is 0.05. The analysis was performed using MINTAB 14. Table 3 shows the result of ANOVA for MA1 and MA2. In Table 3, the p-values of MA1 and MA2 are less than 0.05; therefore, we conclude that there are *significant differences* in selecting modules with different strategies for MAs.

Through the experiment, we demonstrated that two MAs selected stable and reputable modules based on the platform's strategy and reputation mechanisms in a dynamic e-market environment that was represented by DAs' alternative design modules. We expect that the proposed MAS can provide an appropriate method to determine modules for a platform that can be adapted to various dynamic e-market environments.

Table 3:Results of ANOVA for MA1 and MA2

Agent	Source	DF	SS	MS	F-value	P-value
MA1	Factor	3	22048	7349	7.47	0.00
	Error	76	74792	984		
	Total	79	96840			
MA2	Factor	3	29165	9722	12.37	0.00
	Error	76	59731	786		
	Total	79	88896			

6. CLOSING REMARKS AND FUTURE WORK

In a dynamic electronic market environment, a successful product family depends on how to determine a platform that remains stable despite market variations. A platform module can be developed from alternative design methods or provided by different suppliers. Based on this concept, we modeled a multi-agent environment as an e-market consisting of economically-motivated agents to explain agents' behaviors and roles.

A multi-agent system is an appropriate tool to design and implement a product development system in a distributed environment because of its flexibility, scalability, and adaptability. This paper has introduced a multi-agent system based on market mechanisms for module-based product family design. The agent architecture for the MAS was described, including each agent's specific roles, knowledge, and strategies. In this paper, a reputation mechanism was used to select stable and reputable modules for the platform by detecting and dismissing disreputable modules in a dynamic electronic market environment. We have implemented the proposed MAS using JADE and JARE and demonstrated how to use module reputation for selecting a module based on a platform's strategy and reputation mechanisms in a dynamic e-market environment comprised of alternative design modules.

Using the MAS, we can design a module for the platform that can be adapted to various dynamic e-market environments. Therefore, we expect that the proposed system can help to design a variety of products that are defined as a module-based product family. Because negotiation processes in the MAS are developed based on auction mechanisms, we can apply the MAS to a product development system in a supply chain environment. Future research efforts will focus on improving the efficiency and effectiveness of the MAS, enhancing the design agent's knowledge to better reflect different market environments, and expanding its application to web-based product family design.

ACKNOWLEDGMENTS

This work was funded by the National Science Foundation through Grant No. IIS-0325402. Any opinions, findings, and conclusions or recommendations presented in this paper are those of the authors and do not necessarily reflect the views of the National Science Foundation.

REFERENCES

[1] Bakos, J. Y., 1997, "Reducing Buyer Search Costs: Implications for Electronic Marketplaces," *Management Science*, vol. **43**, No. 12, pp. 1676-1692.

[2] Padovan, B., Sackmann, S., Eymann, T., and Pippow, I., 2002, "A Prototype for an Agent-Based Secure Electronic Marketplace Including Reputation-Tracking Mechanisms," *International Journal of Electronic Commerce*, vol. **6**, No. 4, pp. 93-113.

[3] Liang, W. Y. and Huang, C. C., 2002, "The agent-based collaboration information system of product development," *International Journal of Information Management*, vol. **22**, No. 3, pp. 211-224.

[4] Muffatto, M. and Roveda, M., 2000, "Developing Product Platforms: Analysis of the Development Process," *Technovation*, vol. **20**, No. 11, pp. 617-630.

[5] Simpson, T. W., Siddique, Z., and Jiao, J., 2005, *Product Platform and Product Family Design: Methods and Applications*. New York, NY: Springer.

[6] Lee, Y. H., Kumara, S. R. T., and Chatterjee, K., 2003, "Multiagent Based Dynamic Resource Scheduling for Distributed Multiple Projects using a Market Mechanism," *Journal of Intelligent Manufacturing*, vol. **14**, No. 5, pp. 471-484.

[7] Blecker, T., Friedrich, G., Kaluza, B., Abdelkafi, N., and Kreutler, G., 2005, *Information and Management Systems for Product Customization*. New York, NY: Springer Science Business Media Inc.

[8] Madhusudan, T., 2005, "An Agent-based Approach for Coordinating Product Design Workflows," *Computers in Industry*, vol. **56**, No. 3, pp. 235-259.

[9] Jia, H. Z., Ong, S. K., Fuh, J. Y. H., Zhang, Y. F., and Nee, A. Y. C., 2004, "An Adaptive and Upgradeable Agent-based System for Coordinated Product Development and Manufacture," *Robotics and Computer Integrated Manufacturing*, vol. **20**, No. 2, pp. 79-90.

[10] Anumba, C. J., Ren, Z., Thorpe, A., Ugwu, O. O., and Newnham, L., 2003, "Negotiation within a Multi-agent System for the Collaborative Design of Light Industrial Buildings," *Advances in Engineering Software*, vol. **34**, No. 7, pp. 389-401.

[11] Tan, G. W., Hayes, C. C., and Shaw, M., 1996, "An Intelligent-Agent Framework for Concurrent Product Design and Planning," *IEEE Transactions on Engineering Management*, vol. **43**, No. 3, pp. 297-306.

[12] Tran, T. and Cohen, R., 2002, "A Reputation-Oriented Reinforcement Learning Strategy for Agents in Electronic Marketplaces," *Computational Intelligence*, vol. **18**, No. 4, pp. 550-565.

[13] Zacharia, G., Evgeniou, T., Moukas, A., Boufounos, P., and Maes, P., 2001, "Economics of Dynamic Pricing in a Reputation Brokered Agent Mediated Marketplace," *Electronic Commerce Research*, vol. **1**, No. 2, pp. 85-100.

[14] Pine, B. J., II, 1993, *Mass Customization: The New Frontier in Business Competition*. Boston, MA: Harvard Business School Press.

[15] Simpson, T. W., Maier, J. R. A., and Mistree, F., 2001, "Product Platform Design: Method and Application," *Research in Engineering Design*, vol. **13**, No. 1, pp. 2-22.

[16] Moore, W. L., Louvier, J. J., and Verma, R., 1999, "Using Conjoint Analysis to Help Design Product Platforms," *Journal of product innovation management*, vol. **16**, No. 1, pp. 27-39.

[17] Siddique, Z. and W., R. D., 2000, "Product Family Configuration Reasoning using Discrete Design Spaces," *ASME Design Engineering Technical Conferences Proceedings*, Baltimore, MD, ASME, Paper No. DETC00/DTM-14666.

[18] Gonzalez-Zugasti, J. P., Otto, K. N., and Baker, J. D., 2000, "A Method for Architecting Product Platforms," *Research in Engineering Design*, vol. **12**, No. 2, pp. 61-72.

[19] Rai, R. and Allada, V., 2003, "Modular Product Family Design: Agent-based Pareto-optimization and Quality Loss Function-based Post-optimal Analysis," *International Journal of Production Research*, vol. **41**, No. 17, pp. 4075-4098.

[20] Kamrani, A. K. and Salhieh, S. M., 2000, *Product Design for Modularity*. Boston, MA: Kluwer Academic Publishers.

[21] Tarasewich, P. and Nair, S. K., 2001, "Designer-Moderated Product Design," *IEEE Transactions on Engineering Management*, vol. **48**, No. 2, pp. 175-188.

[22] Ulrich, K. T. and Eppinger, S. D., 2000, *Product Design and Development, 2nd*. Boston, MA: McGraw-Hill/Irwin.

[23] Wooldridge, M., 1996, *An Introduction to Multiagent Systems*. England: John Wiley & Sons Inc.

[24] Chandra, C. and Kamrani, A. K., 2003, "Knowledge Management for Consumer-Focused Product Design," *Journal of Intelligent Manufacturing*, vol. **14**, No. 6, pp. 557-580.

[25] Moon, S. K., Kumara, S. R. T., and Simpson, T. W., 2005, "Knowledge Representation for Product Design Using Techspecs Concept Ontology," *IEEE International Conference on Information Reuse and Integration*, Las Vegas, NV, pp. 241-246.

[26] Yu, B. and Singh, M. P., 2000, "A Social Mechanism for Reputation Management in Electronic Communities," *4th International Workshop on Cooperative Information*, Springer-Verlag, Berlin, pp. 154-165.