Electronic Commerce Research, 3: 365–395 (2003)

Autonomous Mapping of E-Business Demands and Supplies via Invisible Internet Agents

HANH PHAM
State University of New York at New Paltz

YIMING YE
IBM, T.J. Watson Research Center

VIEN NGUYEN

Abstract

In business Internet-based systems demands and supplies can be hidden in different forms and locations. We develop a simple type of agents called knowledgeable objects for carrying and mapping heterogeneous and distributed business data in a self-processing way. Internet agents coordinate these invisible agent objects to implement a competitive mapping via agent bidding. Interval-valued numbers and fuzzy ranking are utilized for representing and clustering dynamic business data. The concept of agent awareness and invisibility is used for regulating agent interaction scope to decrease mapping time and to accord with system capacity. Our analysis shows that this combination of agent-interval-based techniques not only meets the distributed, heterogeneous, and dynamic tendencies of E-business systems but also makes the mapping process more autonomous and efficient.

Keywords: Internet agents, business automation, multi-value mapping, interval-valued numbers, fuzzy logics

1. Introduction

The Internet provides opportunities to build large-scale, heterogeneous, dynamic, and joined business systems. However, in this information age, business involves huge amounts of distributed and heterogeneous data which may be visible or invisible on the Internet. In this e-business environment, data processing and many decision making processes need to be *automated* as retrieving, analyzing, and processing these data manually could be extremely costly and time consuming [Horn, 22]. Mapping business demands and supplies into transactions is one of the key steps in business automation. There is an *increasing need* to have efficient mapping models and mechanisms since the business demands and supplies can have various forms, relate to different categories, and participate in different transactions simultaneously.

The problem of mapping demands and supplies in e-business systems may appear in different contexts, such as mapping producers to consumers in e-supply chains [Walsh and Wellman, 57], mapping traders in e-marketplaces [Maes et al., 35; Dailianas et al., 12; Strobel, 54], bids to asks in auctions [Kalagnanam et al., 26], mapping business components to each other in e-management [Huhns and Singh, 23; Sycara et al., 55;

Sheth, 50], mapping applicants to positions in e-recruitment [Gates and Nissen, 17]. Regardless of its contexts business mapping involves: (i) frequent and cooperative usage of distributed and heterogeneous data or knowledge from files, databases, interaction with users, and assisting software, (ii) multiple criteria in estimating e-business values, (iii) dynamic changes of business data, and (iv) possibilities of having incomplete data because the data is not fully available, missing, or damaged.

Mapping business demands and supplies in this environment is extremely difficult since a demand or supply can be involved in several transactions at a time or relate and depend on each other. Besides, the business values of demands and supplies can be evaluated in different ways using different resources. For example, a business partner can be estimated by different criteria such as volume of production, product variety, returned capital, profitability. The estimation from each of these points of views can be made based on various knowledge and data which are distributed at different locations on the networks and with different accreditation levels. Thus, in e-business the classical problem of mapping [Heckbert, 20] now obtains a new and more complicated form than just matching members of a set to the other members of this set. This new form challenges us by the multiple values of multiple criteria, dynamic changes, and the possible incompleteness of data. To deal with these issues we need to be able to: (i) represent and carry business information about demands and supplies given in heterogeneous forms, with frequent changes and different levels of endorsement and completeness, (ii) measure the mapping degrees with multiple criteria and multiple values, and (iii) autonomously coordinate nested and parallel processes with distributed data.

In order to achieve these goals we propose an agent-based and interval-valued model, called ASIAM, which uses: an agent-based tool for data representation called *knowledgeable objects* for handling (i); interval-valued numbers and fuzzy ranking for (ii); and various types of agents as system components to deal with (iii).

We describe knowledgeable objects as invisible agents for representing business demands and supplies and their states in the context of business automation in Section 2. The modeling of E-business values using interval-valued numbers is explained in Section 3. Then, the interval-based fuzzy mapping mechanism is presented in Section 4. The agent-based architecture for coordinating and mapping knowledgeable objects is depicted in Section 5. In Section 6, we describe the e-city project as an environment and case study for the proposed model. Several experimental results on ASIAM in the e-city are provided in Section 7. In Section 8, ASIAM is analyzed via comparisons with related works. Finally, we conclude and discuss future research in Section 9.

2. Invisible agents for representing business demands and supplies

In business automation software agents are built to replace or support intellectual labors in inputting, carrying, and analyzing data, responding to customers, or executing administrative works. The research and developments of software agents have shown their autonomy and flexibility with many useful applications for specific scenarios in different areas such as: coordination [Sycara et al., 55], information management [Bergamaschi and

Beneventano, 8], web search [Somlo and Howe, 52], negotiation [Kraus and Lehmann, 31], auction [Stone et al., 53], trading [Maes et al., 35], automated decision making [Sandhom, 48], contract forming [Collins et al., 10], personal use [Abu-Hakima et al., 1], and user-interface [Lieberman et al., 34]. Therefore, we will adopt the agent concept in building tools for representing the business demands and supplies in a self-processing way as in large-scale business systems the demands and supplies can appear in different forms and locations and it requires a flexible and autonomous means to process them.

Among the software agents used in e-business systems on the Internets only personal agents and user-interface agents are visible to the users while most of other agents are not aware by the users. These agents are invisible because they can run autonomously without user interference and there is no need for the users to monitor them. The *invisibility can also hide the complexity of the systems* from the participants. We will deploy this feature in developing tools for representing business data, especially the demands and supplies. We describe the basic concepts and structures of these agent-based tools called knowledgeable objects, the features of demands and supplies in the business automation context, and how the knowledgeable objects can be used to represent business demands and supplies as follows.

2.1. *Knowledgeable objects as invisible agents*

The knowledgeable objects (KO) are a mix of objects and agents. Implemented as agents they are light-weight autonomous programs. Constructed as objects they have a common and simple structure of three parts: head, data, and instructions. A knowledgeable object is created as an instance of this common prototype where the formats of the data and the instructions are fixed, however the contents of the data and the instructions can be customized. Depending on the permission setting a KO can be open or closed with different degrees to the other KOs. The data of a KO can be processed by the instructions from the other KOs and vice verse. Each knowledgeable object is designed for representing a demand, a supply, or an intermediary transaction component in e-business systems. Business knowledge is integrated into KO through the KO instructions or into the knowledge bases which are provided to the KOs via system services.

In order to reduce the complexity of large-scale business systems, where there may be millions of demands and supplies, we build KO as invisible agents which are not only invisible from the users but may also be *invisible from each other* under given circumstances. This feature is based on the fact that in the real life a person with some demand would restrict himself in small groups instead of going around and ask all other persons in a large society for a match to his demand because of the deadlines and the privacy. In order to reduce the system workload these agents—KO should be light-weight as the number of them can be very high since we need one KO per demand, supply, or data item.

As business demands and supplies come from data items hidden in different forms, a knowledgeable object is used to *represent* an data item of different types: files (text, binary, audio, video, images, etc.), a record or a field in a database, e-mails, data streams from robot sensors, etc. The *head part* contains the KO identification info, links or pointers

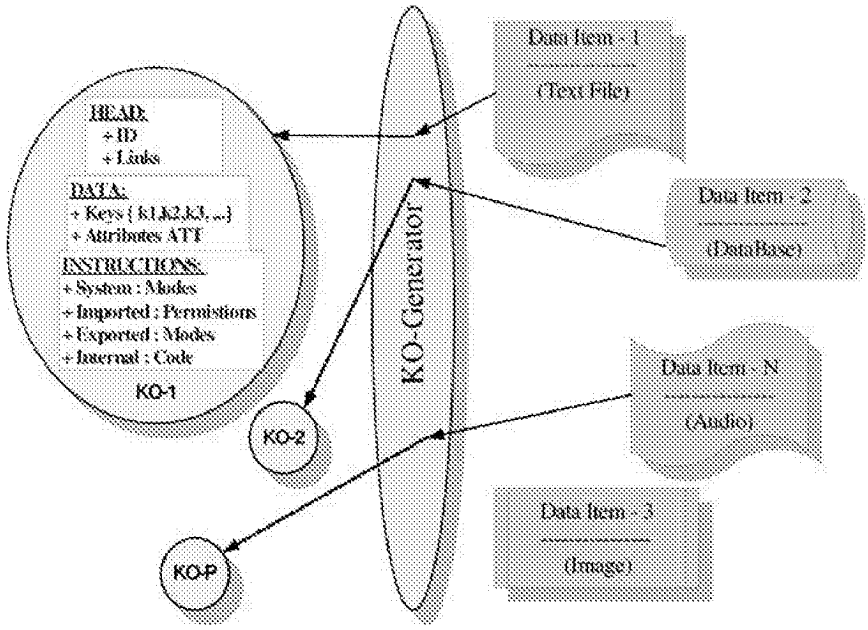


Figure 1. Knowledgeable objects for representing business data.

to the given data item. The *data part* contains the list of keys and other attributes for the given data item such as priority, creation time. The *instruction part* defines the actions which can be implemented over the given data item. There are several types of instructions: system, imported, exported, and internal. System instructions are the instructions defined in the system instruction set (SIS) by the middleware to support KOs. Imported instructions are the instructions defined by other KOs which interact with the given KO. Exported instructions are the instructions defined by given KO which it would like to perform over the other KOs when they interact each other. Internal instructions are the instructions defined by the KO and are implemented when the KO is active. There are permissions for each of these instruction types and they are defined by the owner of the KO when the KO is created. The permissions can be changed during KO life. Figure 1 illustrates the structure of KOs and how they are created from different data items.

Compared with traditional ways of data representation such as files or databases KO represent data items in a more active way. While data items in files and database which are passive and can be retrieved and processed only by external processing programs or queries, with KOs the data can process itself through KO activities using their instructions, the related data can find each other via KO interactions, more details on KO features are described in [Pham, 43].

Having a structure as of objects, KOs are *more autonomous* compared with regular objects in CORBA or DCOM, since each KO runs as an independent program and its data is stored in nonvolatile memory. A regular object and all the data it carries would be destroyed if the supporting program or middleware stops. However, a KO, in the passive

state when its program is not running, can be resumed and continued with all the data saved from previous activities. Another difference of KO from regular objects is that even though regular objects may contain member functions they cannot interact with each other by themselves directly without additional instructions in the programs which harbor them while KO can, as explained later in Section 2.3 or in [Pham, 43].

Having autonomous features of software agents, the KO model is different from the other agent-based systems by using one KO for each single data item as a combined *representation-processing tool* with a one-to-one style. Most other models build an agent for processing many data items, as a *processing tool* rather than as a representation tool, and with a one-to-many style. Thus, a data item represented and processed by one KO may be more active than the data items many of those are processed by one agent. Nevertheless, the disadvantage of KO model would be the high number of KOs which would require much more system resource compared with the other agent-based systems. That is why we propose the agent invisibility concept to reduce the scope of KO interactions and use the agent passive state to decrease the resource usage. More details on how the agent invisibility and awareness concept helps to regulate the scope of KO interactions is described in Section 2.3.

2.2. *Demands and supplies in business atomation*

Conducting business requires business data BD and business knowledge BK. The business data BD provides information about: (i) the states of business parties which participate in different business processes or transactions, and (ii) the states of business transactions $T = \{T_1, T_2, \ldots\}$. The goals of business parties can be divided into demands $D = \{D_1, D_2, \ldots\}$ and supplies $S = \{S_1, S_2, \ldots\}$. A match of supplies to demands $M(\{\check{D}\}, \{\check{S}\})$, where $M : D, S \rightarrow T$, $\check{D} \in D$, $\check{S} \in S$, can result in a business transaction $T^* \in T$. The business knowledge BK contains the rules for business transactions defining when or how they should occur. Business activities A are needed to obtain, update, and prepare data for the transactions. Business activities A can be considered as internal business transactions, i.e. $A \subset T$.

The goal of business automation is to reduce human participation to the minimum so that it would decrease operation costs, time delays and therefore increase profits and the quality of service. Business automation includes the automations of obtaining data, generating business demands and supplies, mapping demands and supplies into transactions and carrying out these transactions on behalf of business parties. Examples of autonomous or partly autonomous business systems are amazon.com, e-bay.com, travelocity.com. Not only a purchasing order but a request for a meeting with the manager can also be a business demand. In this case the availability of the manager at a specific time can be considered as a business supply.

The intricacies of business automation are that business data DB can have different forms, come from different sources, available at different times and relate to each other in different ways. Besides, regardless of the automation degree there are always some steps that may involve human actions such as users in making business demands or experts

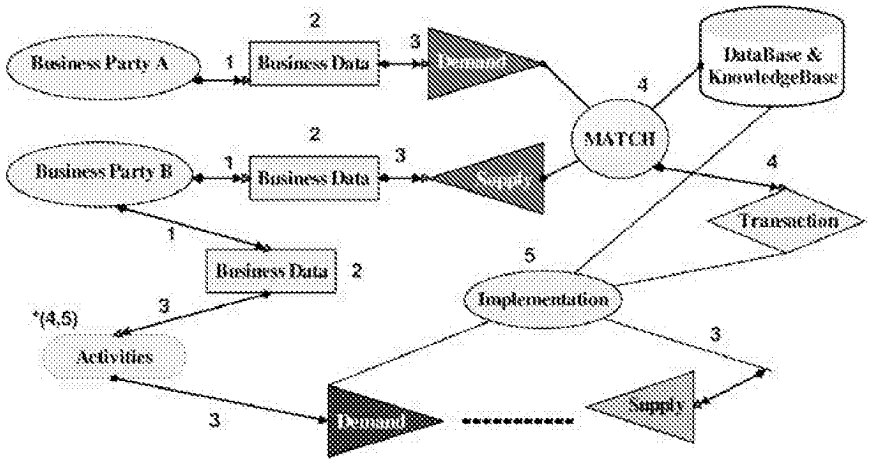


Figure 2. Demands and supplies in an automation business scenario.

and managers in making the final decisions. Therefore, the business automated systems should also be able to integrate human activities. In order to build a business automation system we would need to do the following:

1. Automation of data retrieval, updating, and integrating (Data Automation).
2. Automation of data analysis and linking (Analysis Automation).
3. Automation of formulating demands and supplies (Demand–Supply Automation).
4. Automation of forming transaction (Transaction Automation).
5. Automation of carrying out transactions (Implementation Automation).

An example of how demands and supplies involve in these automation steps is illustrated in Figure 2.

Data Automation. Retrieve, update, and integrate business data automatically is a big challenger since data can be heterogeneous and distributed coming from: (i) data sources such as databases, files, Internet, etc., or (ii) interactions with people such as on-line dialogues, phones, faxes, etc. There are two main approaches in dealing with this problem: structural and semantic [Bergamaschi and Beneventano, 8]. While structural methods require intermediate brokers semantic methods require the acceptance and the use of the same standards in describing and creating data for all sources.

Analysis Automation. Analyze and link automatically data which relate to each other is another difficult problem. Techniques developed for data mining and data warehousing [Kargupta et al., 28] and knowledge discovering [Jensen et al., 25] can be used for this step of business automation.

Demand–Supply Automation. The automation of formulating business demands and supplies means to generate business requests when they are needed and to represent business

capabilities of business resources when they are available, all without human intervention. For example: an invoice attached to an e-mail received by the purchase department should generate automatically (without having people to read the e-mail or open files) a *business demand* to the accounting department for making the proper corresponding payment; on the other hand, as long as the balance allows and the purchasing rules are satisfied the capability of the accounting department to make such payments should be represented as a *business supply* and should be aware of by all other departments. As most of the existing systems are specialized in specific problems, this automation step is done by the user-interface component which directly accepts the demands and supplies from the users in specific formats [Sen et al., 49]. The existing mechanisms work well in specific domains but we lack general-purpose mechanisms which can generate heterogeneous demands and supplies from heterogeneous business data.

Transaction Automation. In order to establish the transactions automatically we need to match business supplies to business demands [Pham and Sharif, 46]. Matching could have forms of not only one-to-one but also one-to-many or many-to-one [Dailianas et al., 12], or it could take a chain of supplies to satisfy a demand. A transaction may includes or triggers other transactions such as in supply chains [Walsh and Wellman, 57] or B2B. Once, when the demands and supplies are matched they usually should be eliminated from further matching unless the transaction is cancelled. However, in some cases the supplies which represent almost unlimited capabilities should still be kept in the system after the matching. The key point of successful business automation is how to formalize the demands, supplies, and the transactions so that it is universal while reflecting precisely different business logics in the real world.

Implementation Automation. The automation of carrying out separate transactions is straightforward. Everything such as terms or conditions is already given in the formulated transactions. The procedure of how to implement a given transaction is defined based on the rules given in the business knowledge-bases and business databases. However, implementing related transactions is more sophisticated [Sheth, 50] as it requires coordination and conflicts solving.

The Demand–Supply Automation and the Transaction Automation steps play a central role in business automation and they may have significant impact on the overall performance of the e-business system. In the next section we will describe how knowledgeable objects can be used to make the formulation and mapping of business demands and supplies into transactions autonomous and efficient.

2.3. *Representation of business demands and supplies via knowledgeable objects*

In the current e-business systems, data about business components such as money, business parties, activities, and transactions may be stored in database but when the business data are hidden primarily in document files, e-mails, etc. we would need to retrieve and then

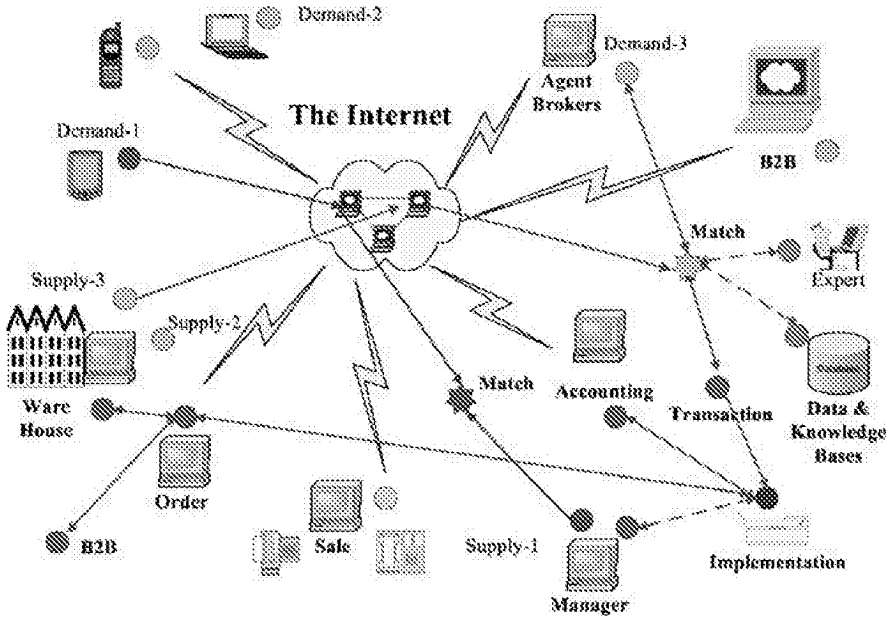


Figure 3. Business demands/supplies, transactions, and implementations with knowledgeable objects.

input them into the database before having them processed by other application programs. The KOs can be used to reduce the overheads of this preprocessing step. Since a KO can be generated to represent data items in different formats as explained in Section 2.1, we can use KOs to represent the business demands and supplies as follows. First, the definitions of demands or supplies are defined based on the business logistics and the contexts. The requirements for a KO to become a demand or supply are based on the values of its data which include its keys and its attributes. For instance, an e-mail represented by a KO with $K_d = \{k_d1, k_d2, \ldots\}$ and attributes $\text{ATT}_d = \{A_d1, A_d2, \ldots\}$ (Figure 1) is a demand D^* to see Ms.X from Mr.Y if the words "Ms.X", "see" or "meet" or "appointment", etc. can be found in K_d with the orders, frequencies, and structure characteristics shown in the attributes from ATT satisfied some given thresholds. Meanwhile, a KO which represents a schedule in the PDA of Ms.X can turn into a supply S^* to make an appointment with Mr.Y if its keys K_s and attributes ATT_s satisfy some given corresponding requirements. Hence, the knowledgeable objects can be used to represent business data in general which will become business demands or supplies when their data achieve some given states defined for specific business activities and transactions. Since KOs are independent and autonomous programs they are suitable for representing distributed business data which may contain demands or supplies at different locations in a distributed large-scale business system as shown in the scenario above. An example of using knowledgeable objects for representing demands and supplies among other business components is shown in Figure 3.

The procedure of business automation which includes the formulation of demands and supplies using KOs can be formulated as the following:

Given: Q business objects: $O = \{O_1, O_2, \ldots, O_Q\}$; a set XS of states; a set of system operations OO; a set IR of self-interaction rules for KOs; a definition Dfd of demands; a definition Dfs of supplies; a definition Dft of transactions; a set FR of transaction formation rules for demands and supplies; a set RR of firing rules for implementing transactions.

Business automation cycle consists of:

- Data Automation: Generate and update $O_i \in O = \{O_1, O_2, \ldots, O_Q\}, i = 1, \ldots, Q$,
- Analysis Automation: Let $O = \{O_1, O_2, \ldots, O_Q\} \rightarrow [\text{IR}] \rightarrow O = \{O_{1'}, O_{2'}, \ldots, O_{Q'}\}$,
- D–S Automation: Define $O = \{O_{1'}, O_{2'}, \ldots, O_{Q'}\} \rightarrow [\text{Dfd, Dfs}] \rightarrow \{D_1, D_2, \ldots, D_H\} \cup \{S_1, S_2, \ldots, S_K\}$,
- Transaction Automation: $\{D_1, D_2, \ldots, D_H\} \cup \{S_1, S_2, \ldots, S_K\} \rightarrow [\text{Dft}] \rightarrow \{\text{Tr}_1, \text{Tr}_2, \ldots, \text{Tr}_R\}$,
- Implementation Automation: $\text{Tr}_1 \rightarrow [\text{RR}] \rightarrow \{O_{1,1}, \ldots, O_{1,m1}\}, \ldots, \text{Tr}_R \rightarrow [\text{RR}] \rightarrow \{O_{R,1}, \ldots, O_{M,mR}\}$,

where O_i is a knowledgeable object, D_i is a demand, S_i is a supply, Tr_i is a transaction and $A \rightarrow [R] \rightarrow B$ means a transformation from A to B by the rules given in R.

Thus, business data represented by KOs are transformed through the changes of KO states and features. The demands $\{D_1, D_2, \ldots, D_H\}$ and supplies $\{S_1, S_2, \ldots, S_K\}$ are actually hidden in the knowledgeable objects $\{O_1, O_2, \ldots, O_Q\}$ before they are discovered by the rules given in Dfd and Dfs. In other words, the formulation of demands and supplies is carried out via the interaction of KOs which change the states and features of KOs. Once the states and the features of KOs satisfy specific requirements defined in Dfd and Dfs the given KOs are recognized as the representatives of the corresponding business demands or supplies as $\{D_1, D_2, \ldots, D_H\}$ and $\{S_1, S_2, \ldots, S_K\}$. A KO can represent a business data item which may contain at the same time several demands or supplies in different transactions simultaneously. The KOs run within a middleware whose components and architecture will be depicted through the next sections. The KO states, operations, and interactions are the foundation for forming the demands and supplies. They can be described as the follows.

States and operations of knowledgeable objects KO may have the following states XS = {X-new, X-act, X-bus, X-pas, X-ter} stands for new, active, busy, passive, and terminated states. When an object O_i is in the active state it can implement its internal instructions IR and interact with the other object O_j by implementing exported instructions from O_j, and let object O_j implement the exported instructions of O_i.

In the context of business automation, the system has the following operations on business objects $\{O_1, O_2, \ldots, O_Q\}$: {Creation; Activation; DisActivation; Interaction; Termination} = {CrO, AcO, DiO, InO, TeO}. The Creation operation CrO(BD) generates a new KO for a data item based on the creation rules CR : {BD, BK} $\rightarrow$ [CR] $\rightarrow$ $O_i \in O = \{O_1, O_2, \ldots, O_Q\}$. The newly created KO is put in the KO-base of BKOAS in a non-volatile memory. The Activation operation AcO(O) on an object loads the given object to the KO-pool in the main memory which contains the active objects. Once the object is in the KO-pool it runs as a thread. The DisActivation operation DiO(O) moves

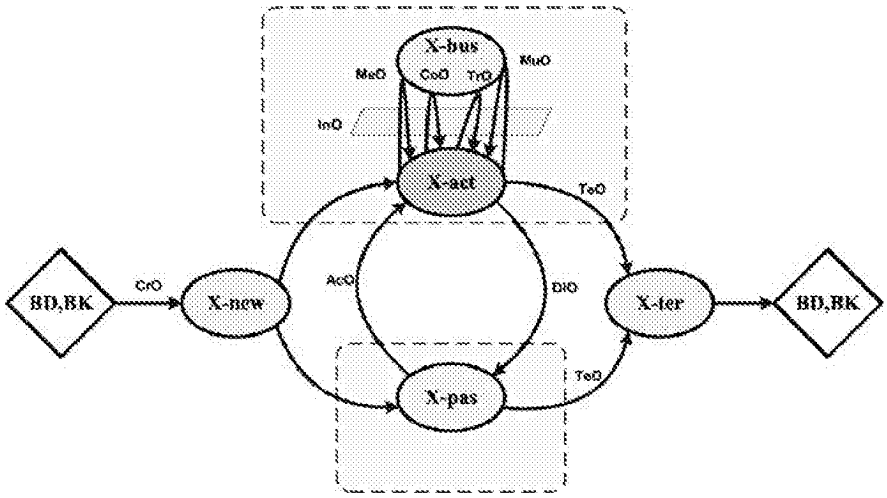


Figure 4. States and operations of KOs.

an object back to the KO-base from the KO-pool. The Interaction Operation InO(O) sets the given object ready for interaction with other objects. The Termination operation TeO ends KO and may produce the corresponding data or knowledge based on the creation rules TR: $O \rightarrow$ [CR] $\rightarrow$ {BD, BK}. Figure 4 shows how the states of KO are transformed from one to another and which operation is involves in those transformations.

KO interactions While in the X-act active state an object O_i runs its internal instructions and is waiting for a system operation or an interaction with the other objects: <X-act> = {<Running>, <Interaction>}. In the <Running> mode, the object O_i can be set to one of the following: {<Waiting>, <Self-Transformation>}. In <Waiting> it updates the keys $\{k1, k2, \ldots\}$ based on the changes in the data item DT_i. In <Self-Transformation> it may change the data and the info in the object itself depending on the its internal instructions. In the <Interaction> mode, the object may be set to one of the following sub-I-operations: {Merge, Exchange, Multiplication}. These sub-operations are carried out under the Interaction operation InO as described in Figure 5.

In the Merge mode: the merge operation MeO combines the given object with other active objects based on the merge rules MeR which define the conditions of the kinds of objects and in which situation can be merged, MeO: $O \rightarrow$ [MeR] $\rightarrow O$, new objects may appear after merging. In the Multiplication mode: the multiplication operation MuO multiplies an active object into a group of the same objects, based on the multiplication rules MuR, which may require the presence of some other objects as the stimulant of the multiplication, MuO: $O \rightarrow$ [MuR] $\rightarrow O$. In the Exchange mode: the Exchange operation ExO let the given object O_i to implement the exported instructions from other object O_j and vice verse based on the Exchange rules ExR, ExO: $O \rightarrow$ [ExR] $\rightarrow O$.

For instance, a data item Data-k contains a business *demand* D-k and a data item Data-h contains a business *supply* D-h. The corresponding objects which are generated are O_k

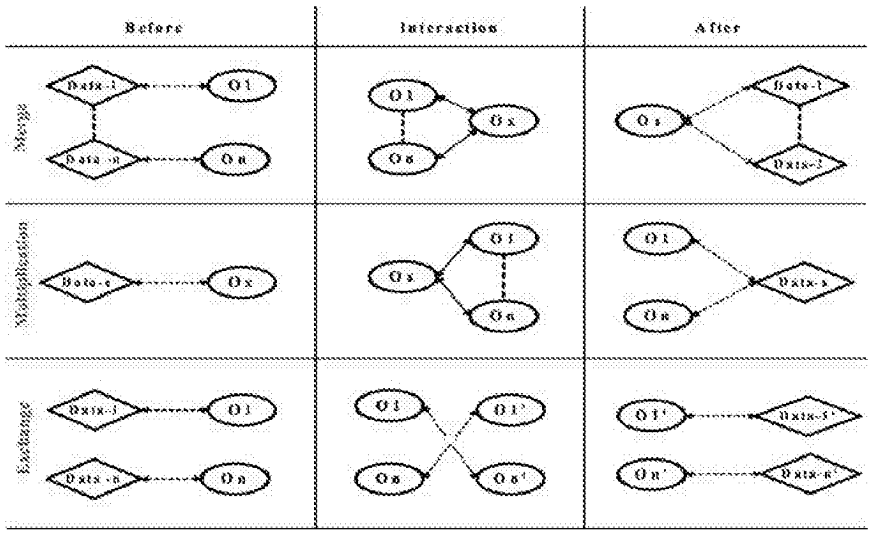


Figure 5. I-operations for KO interaction.

and O_h. If O_k and O_h satisfy the transaction rules then O_k and O_h will result a transaction when they interact with each other through one of the I-operations. We may have $\{O_k, O_h\} \rightarrow [\text{MeR, TR}] \rightarrow O_t$ where O_t is an object representing the transaction. If nothing else is needed then the transaction is implemented by firing the given object $O_t : O_t \rightarrow [\text{RR}] \rightarrow$ Data-T to produce the necessary data Data-T which specifies that the transaction is completed. The firing is done either via the termination operation TeO or via the <self-transformation> in the <Running> mode.

In the dynamic cases when there may appear a new or additional requirement, i.e. the TR is changed, the new requirement then is represented as an object O_s. Once the requirement is satisfied or O_t is fired it will change the features of object O_s so that O_t and O_s can form an additional transaction, then we have $\{O_t, O_i\} \rightarrow [\text{MeR, TR}] \rightarrow O_{t'}$ and so far. If we use $\Xi(O_x, O_y)|\{R\}$ as a notation of the outcome of the interaction between O_x and O_y under the rule R then we have: $\Xi(O_k, O_h)|\{\text{Mer, TR}\} = O_t$ and $\Xi(O_t, O_s)|\{\text{Mer, TR}\} = O_{t'}$.

KO awareness and invisibility The concepts of the awareness and the invisibility of KOs are needed to reduce the scope of KO interactions. When the business system is large and distributed the number of KOs and the communication overheads could be very high. If each KO is able to interact with all other KOs in the system then the time a KO may need to interact with all other KOs to select its match could be much more than the time constraint of the request represented by the given KO. In this case, a system of $N = 1{,}000{,}000$ of KOs would have to handle $10^7 \cdot (10^7 - 1)/2$ or more than 10^{13} interactions at a time! Therefore, it is essential that the system can be able to control and adjust the number of the interactions at a time based on the currently available resources.

In order to achieve this goal we add a characteristic called awareness to the KOs. A KO can interact with the other KOs only if they are *aware* or *visible* of those objects. If a KO

is not aware by the other KOs then it is *invisible* to the others. In more details, this concept can be used to regulate the interaction scope of the KOs by the following rules:

Awareness–Interaction Principles:

- If two objects O_i and O_j are invisible (when they are not aware of each other) they cannot interact with each other:

  AIP-1: $O_i !\diamond O_j \Rightarrow \Xi(O_i, O_j)|R = \varnothing, \quad \forall R.$

- The two objects O_i and O_j are interactive if they are active, aware of each other, and their features satisfy one of the I-operation rules: MeR, MuR, ExR.

  AIP-2: $S(O_i) = \text{X-act}; S(O_i) = \text{X-act}; O_i \diamond O_j$
  $\Rightarrow \Xi(O_i, O_j)|(\text{MeR}\|\text{MuR}\|\text{ExR}) = \text{Io}, \quad \text{Io} \in \{\text{MeO}, \text{MuO}, \text{ExO}\}.$

- In AIP-2: if all MeR, MuR, ExR are satisfied then these operations will be carried out based on their priorities. For example, if ExO has the highest priority among {ExO, MuO, MeO} then the operation ExO will occur:

  AIP-3: $\text{Io}: P(\text{Io}) = \text{MAX}\big(P(\text{MeO}), P(\text{MuO}), P(\text{ExO})\big),$
  where $P(x)$ is the priority function.

- If there are several objects which qualify for the same I-operation with the given object O_i then the partner object will be chosen among the candidates based on object priorities, in the simplest case it could be first-come-first-serve.
- The two objects O_i and O_j are non-interactive or inert if their features do not satisfy any of the I-operation rules: MeR, MuR, ExR. Then, no interactions will occur even if they are aware of each other (see Figure 6).

Thus, the awareness degree defines the interaction scope of objects. By making a number of other KOs invisible from a given KO we can reduce the computation complexity and the work load of the given KO. In ASIAM we build the aware rules based on the smart distance concept proposed in [Ye et al., 59]. The object awareness network OAN is maintained by the KO-services which updates the smart distances between objects in $O = \{O_1, O_2, \ldots, O_Q\}$ through an awareness matrix:

$$\psi(t) = \begin{pmatrix} \delta_{1,1}(t) & \ldots & \delta_{1,Q}(t) \\ \vdots & & \vdots \\ \delta_{Q,1}(t) & \ldots & \delta_{Q,Q}(t) \end{pmatrix}.$$

The objects are aware of each other if their features satisfy the aware rules AR. In the simple case AR can be stated as the following: if the smart distance between objects O_j and O_i at the time moment t is $\delta_{i,j}$ and $\delta_{i,j} < R_{\text{lim}}$ then O_j and O_i are aware of each other: $O_i \diamond O_j$. Thus, an object O_i can "see" a number of objects in its visible zone as shown in Figure 6, the other objects are invisible and therefore will not be included in the candidate list for interaction with O_i. By adjusting the awareness degree R_{lim} we can change the scope of the object interaction.

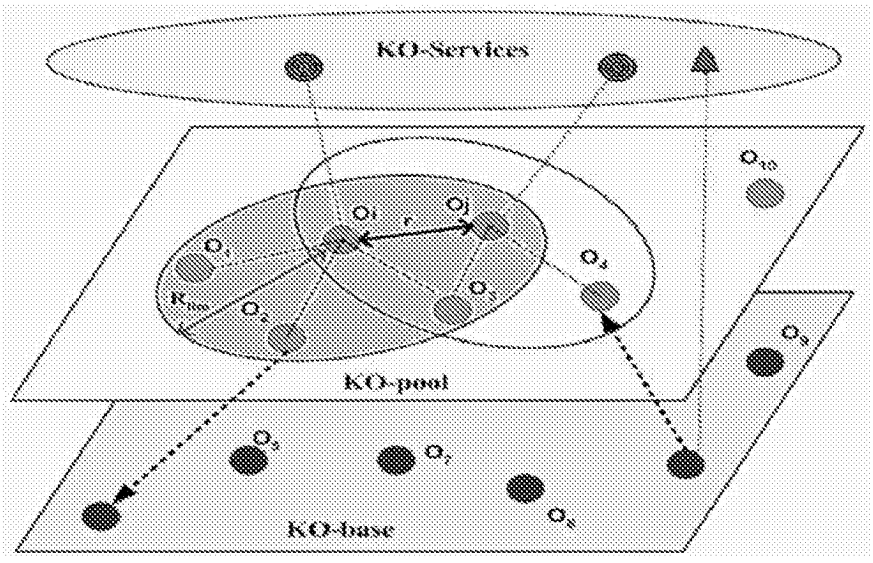


Figure 6. Visible and invisible zones of KOs.

3. Modeling e-business values with intervals

One of the key issues in mapping business demands and supplies is to represent and model their business values. In e-business systems the value of a business component, business object, or transaction can be evaluated by different criteria and by different knowledge sources. Assume that there are X objects to be evaluated by M criteria and by N experts:

- $O = \{O_1, O_2, \ldots, O_X\}$,
- $C = \{C_1, C_2, \ldots, C_M\}$,
- $E = \{E_1, E_2, \ldots, E_N\}$.

Let $\mathrm{Fw}(O_x, C_m, E_n)$, where $O_x \in O$, $x = 1, \ldots, X$, $C_m \in C$, $m = 1, \ldots, M$; $E_n \in E$, $n = 1, \ldots, N$, be the value function which returns a value $w[x, m, n]$ of the object O_x evaluated by the expert E_n and by the criterion C_m. An example of these values is shown in Figure 7 through a 3-D illustration where the coordinators of circles show the sources of evaluation and the circle radiuses show the absolute values for the given object. Experts may use different domains and scales for different criteria. If W is the global domain for business values and W_m^n is a subdomain by criterion C_m given by expert E_n, then we have:

$$(O, C, E) \xrightarrow{\mathrm{Fw}} W = \bigcup_{n=1}^{N} \bigcup_{m=1}^{M} W_m^n.$$

The experts can have different accreditation levels $\mathrm{AE}(m, n)$, $n = 1, \ldots, N$, $m = 1, \ldots, M$, for each field represented by a criterion. For instance, a human specialist may

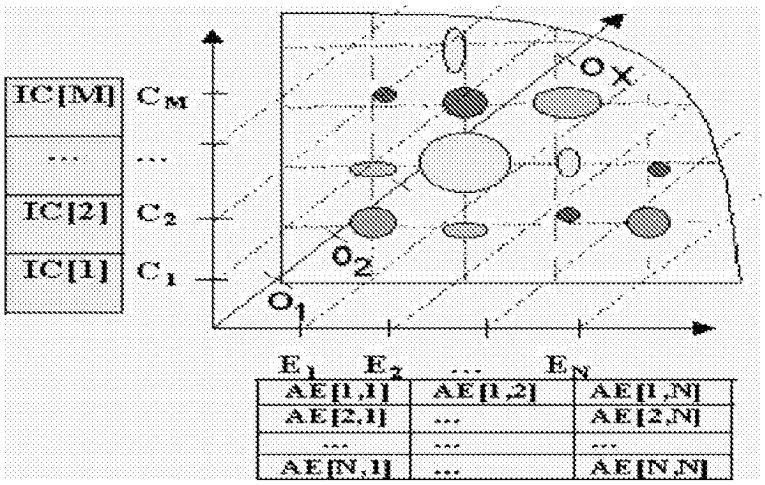


Figure 7. Multi-dimension business values.

have much higher accreditation in evaluating applicants through interviews compared with a software which reads CVs of the applicants. However, a software may have higher accreditation than a human accountant in detecting math errors in structured accounting database or documents such as tax forms.

In evaluating a business object by different criteria, each criterion C_m may have different degree of importance $\mathrm{IC}(m) = k_m$, $m = 1, \ldots, M$. Thus, even by only one criterion, a business object O_x, $x = 1, \ldots, X$, has multiple values: $\{w[x, m, 1], w[x, m, 2], \ldots, w[x, m, N]\}$ given by N experts with different accreditation levels: $\{\mathrm{AE}(m, 1), \mathrm{AE}(m, 2), \ldots, \mathrm{AE}(m, N)\}$. These values are called absolute since experts may use their own measurements and scales. In order to be able to compare these values we need to unify them in the same measurement and scale. We use the following function $\mathrm{Rw}(O_x, C_m, E_n) = v[x, n, m]$ to convert the absolute values into the compatible values:

$$v[x, m, n] = 2\frac{w[x, n, m] - (\mathrm{Max}\{\mathrm{Fw}(O_x, C_m, E)\} + \mathrm{Min}\{\mathrm{Fw}(O_x, C_m, E)\})/2}{(\mathrm{Max}\{\mathrm{Fw}(O_x, C_m, E)\} - \mathrm{Min}\{\mathrm{Fw}(O_x, C_m, E)\})}. \quad (1)$$

The domains are converted as the following:

$$W_m^n|_{n=1,\ldots,N,\, m=1,\ldots,M} \xrightarrow{\mathrm{Rw}} V = [-1, +1].$$

The next step in modeling business values is to represent these multiple values, which are given by different experts, by a single quantity. A traditional and simple method is to use an average value of $\{v[x, m, 1], v[x, m, 2], \ldots, v[x, m, N]\}$ and treat $\{\mathrm{AE}(m, 1), \mathrm{AE}(m, 2), \ldots, \mathrm{AE}(m, N)\}$ as their weights $\{\lambda_1^m, \lambda_2^m, \ldots, \lambda_N^m\}$. However, such approach may loose information as many different distribution of these values may have the same average, for example, $99+1 = 49+51$. Besides, in a dynamic environment when the values and the accreditation levels can be changed frequently if we combine these val-

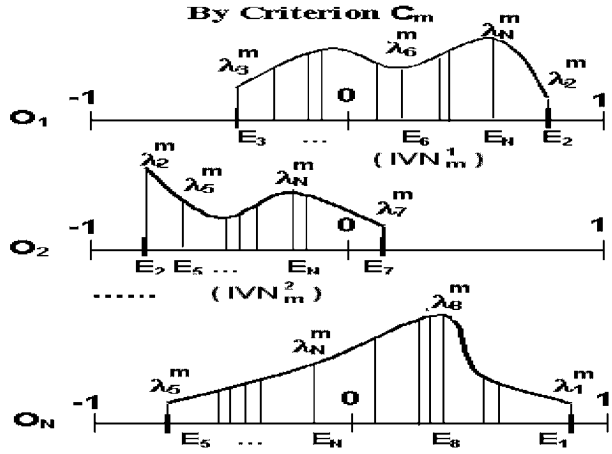


Figure 8. Multi-value business quantity represented by IVN.

ues into one single value immediately. Later, at the mapping moment, this value may not reflect the current situation at that time.

Therefore, we propose to use interval-value numbers (IVN) to represent these multiple values of a quantity. IVN is a special data type which can be built to have a memory and to be accumulative rather than just to show the current value at a given moment [Dubois and Prade, 15; Kreinovich, 32; Pham, 42]. Examples of IVN which represent the business values of object O_x, $x = 1, \ldots, X$, by C_m, is shown in Figure 8 and can be represented as the following:

$$\mathrm{IVN}_m^q = \bigcup_{n=1}^{N} \left(v[q, m, n], \lambda_n^m\right). \tag{2}$$

The use of IVN also allows us to represent the incomplete data. The data incompleteness may be represented by probability or rough intervals, however none of them have memory and ability to accumulate the past values.

The values of business objects are now represented in IVN of the same scale, the same data type, and therefore are compatible. However, each IVN has multiple values with different distributions. We cannot use regular mathematical comparison rules, which work for exact numbers, to compare these INVs. Thus, we will propose a special mechanism to compare IVNs. This mechanism is described in the next section.

4. Interval-valued mapping

The problem of mapping business objects from a set, whose values are represented by IVNs, to the objects in this set can be stated as follows.

Given: (i) X objects $\{O_1, O_2, \ldots, O_X\} = O$; (ii) a set of criteria $C = \{C_1, C_2, \ldots, C_M\}$ whose importance are estimated as $\{k_1, k_2, \ldots, k_M\} = K$, respectively; (iii) each object

O_x has a business value $I(x, m)$ by a criterion C_m, where $x = 1, \ldots, X$, $m = 1, \ldots, M$; (iv) rules, which define what the mapping degree $\mu(O_x \to O_{x^*}, m) = (I(x, m) \oplus I(x^*, m))$ of the transaction between O_x and O_{x^*} should be if $I(x, m)$ maps other $I(x^*, m)$ which is denoted as $O_x \mapsto O_{x^*}$, where $\oplus$ is the mapping operator, which is $+$ in the simple case.

Requirements (one of the following):

- Competitive local mapping: to maximize the mapping degrees of mapped transactions:

$$\forall x\colon \mu(O_x \to O_{x^*}, m)\Big|_{O_x \mapsto O_{x^*}}^{x^* = \{1, \ldots, X\}\backslash x} \to \text{MAX or} > \text{LIM},$$

  where LIM is some given limit.
- Cooperative global mapping: to maximize the number of mapped transactions or to maximize the total degree (values) of all mapped transactions.

$$\sum_{x=1}^{X} \mu(O_x \to O_{x^*}, m)\Big|_{O_x \mapsto O_{x^*}} \to \text{MAX}.$$

- Combined mapping: to have a good balance between the competitive local and the global cooperative mapping.

In this paper we will focus on the local competitive mapping with MAX degrees. Regardless of the requirement type we need to build a measurement system and a mechanism to evaluate the mapping degree $\mu(O_x \to O_{x^*}, m)$ of any two objects O_x and O_{x^*} based on the IVN values: $I(x, m)$ and $I(x^*, m)$, $m = 1, \ldots, M$.

Mapping measurement is a key issue in mapping evaluation. There exit several measurement methods which work with exact numbers: (i) correlation coefficients [Everitt, 16], (ii) distance measures [Everitt, 16], (iii) associate coefficients [Everitt, 16], (iv) probabilistic similarity coefficients [Everitt, 16]. However, these measurement mechanisms cannot sufficiently reflect business relationship represented in IVNs. For example, if 1 means the highest wish to buy, -1 means the highest wish to sell, and 0 means no interest to sell or buy, then if we use (i) or (iii) we have $1 + (-1) = 0.1 + (-0.1)$ which means the result of a business meeting of a person who wants to buy a thing desperately and a person who wants to sell this thing desperately would be the same as the result when two persons with almost no interest to buy or to sell. That is obviously not true. Method (iv) is not practical because we would not have enough information to define the probabilities. Method (ii) work with exact numbers but cannot be applied for IVNs.

Therefore, we propose to define the mapping degree of two objects O_x and O_y which have mapping values δ_x^m and δ_y^m as the following:

$$\text{IMD}_n^m(x, y) = \frac{\text{Max}(|\delta_x^m|, |\delta_y^m|) \cdot \text{Ln}(|\delta_x^m - \delta_y^m| + 1)}{e^{(|\delta_x^m + \delta_y^m| + \text{Sign}(\delta_x^m \times \delta_y^m))}}. \tag{3}$$

Examples of these values are shown in Figure 9 in different cases: opposite-different (1–11); opposite-same (11–22); random (22–40).

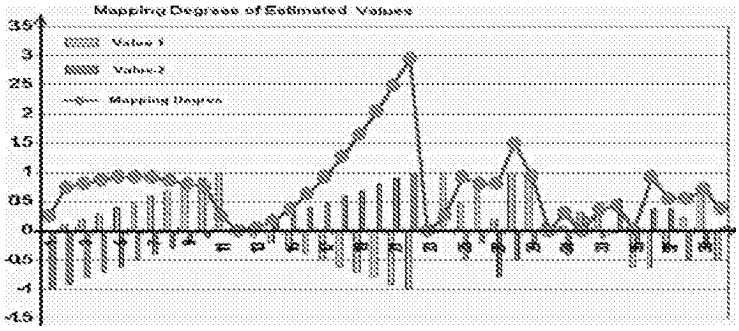


Figure 9. Mapping degrees of IVN.

Then, the combined mapping degree, by all criteria is:

$$R(x, y) = \sum_{m=1}^{M} k^m \cdot \mathrm{IMD}^m(x, y). \tag{4}$$

To compare IVN values we use a modified version of the Density-Based Scheme (DBS) which is based on fuzzy logics and is described in [Pham, 42]. DBS is a special comparison mechanism for general IVNs based on frequency with no peak and when the number of experts N is small (<100). In applying DBS the expert accreditation levels are treated as value frequencies of an IVN. The original Density-Based Scheme in [Pham, 42] returns comparison degrees $D(I_x, I_y) \in [-\infty, +\infty]$ for an IVN pair (I_x, I_y). However, we will use a modified version that returns degrees $D \in [-1, 1]$ to avoid over-bounded numbers. In summary, the competitive mapping mechanism for a set of X IVNs is carried out through the following steps:

1. Define mapping values:
   for $m = 1, \ldots, M$:
     for $x = 1, \ldots, X$:
       define δ_x^m, based on (2);
2. Define mapping degrees:
   for $x = 1, \ldots, X$: (in parallel)
     for $y = 1, \ldots, X \backslash x$: (in parallel)
       for $m = 1, \ldots, M$:
         {for $n = 1, \ldots, N$:
           define $\mathrm{IMD}_n^m(x, y)$ based on (3);
          define $\mathrm{IMD}^m(x, y)$;
         }
3. Compare mapping degrees:
   for $m = 1, \ldots, M$:
     for $x = 1, \ldots, X$: (in parallel)
       for $y = 1, \ldots, X \backslash x$: (in parallel)
         calculate D degrees by DBS for $\mathrm{IMD}^m(x, y)$;

4. Ranking comparison mapping degrees:
   for $x = 1, \ldots, X$: (in parallel)
     for $y = 1, \ldots, X \backslash x$: (in parallel)
       for $m = 1, \ldots, M$:
         $\mathrm{MP}(x) = z$: $R(x, z) \geq R(x, y) | y = 1, \ldots, X^*$ (valid X);
5. The result of mapping is:
   $O_x \mapsto O_{\mathrm{MP}(x)|x=1,\ldots,X}$.

5. Agent-based architecture for automated mapping

In developing an agent-based autonomous mapping system (AMS) we use KOs to represent business data and build other agents for coordinating KOs to map business demands and supplies into matched pairs for transactions. AMS consists of two main parts: (i) middleware KOM which support KOs with general services, and (ii) mapping supporter MS which coordinates the KOs, containing business demands or supplies, to map demands and supplies into transactions. The middleware KOM and the mapping supporter MS can be described as follows.

5.1. *Middleware KOM*

In order to keep the passive objects, to run the active objects, and to arrange object interaction, the middleware KOM has the following components: {KO-generator, KO-Services(Names, Distance), KO-base, KO-pool, System-KnowledgeBase} and has a structure as shown in Figure 10. The KO-generator produces a KO for each data item or request. The KOs are registered and given ID by the KO-Services. Depending on the KO population, KO priorities, and the system capacity KOs are activated in the KO-pool or are turned into passive state and are stored in the KO-base. The KOs can be switched

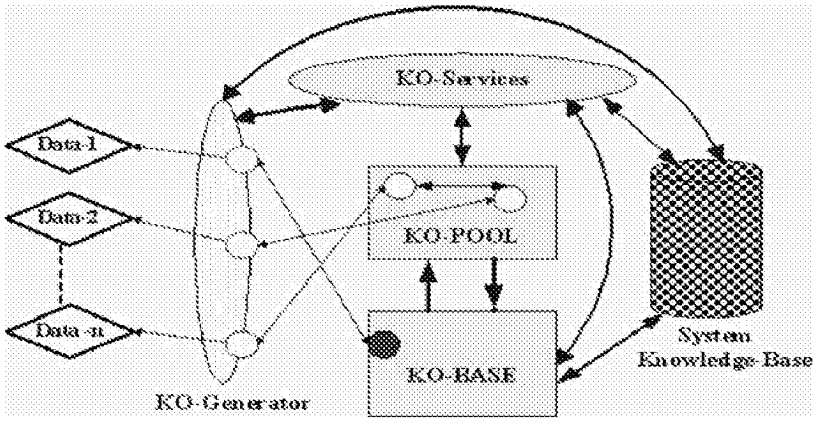


Figure 10. KOM components and architecture.

from the active state in KO-pool into the passive state in the KO-base, and vice versa. The System-KnowledgeBase contains the interaction and transaction rules, and the system instructions for KOs to update their own knowledge. All other components of KOM use the System-KnowledgeBase for defining their working regulations.

The objects are distributed by two layers: KO-pool for the active ones and KO-base for the passive ones as shown in Figure 6. The KOs are divided into clusters based on their distances which are defined by the KO-Services by different criteria. The objects which are aware of each other interact with each other in a peer-to-peer fashion [Minar, 39]. An object can interact only with a given number of KOs at a time defined by {MeR, MuR, ExR}. While it involves with an operation the given object is in the state X-bus. When there are several objects which can serve as candidates for an interaction with the given object these objects compete with each other by the competition rule CoR during a given period of time called time working window TWW.

Protocols:

Checking phase:

1. O_i asks KO-Services for the lists of neighbors LN(O_i).
2. KO-Services defines LN(O_i) based on the smart distances between O_i and the other objects.

Requesting phase:

3. O_i chooses a selected list of KOs it wishes to interact with.
4. O_i sends the requests based on the list and wait for the first arrivred response from a KO, O_j for example.
5. Once a response is received the O_i stops sending requests and sends the confirmation to O_j.

Selection phase:

6. O_j may receive several requests during TWW.
7. It chooses a request among the received requests, from O_i for example, based on its own selection rules.
8. O_j sends a response and waits for a confirmation from O_i.

Interaction phase:

9. O_i sends a confirmation to O_j, save the current state and sets its new state to X-bus.
10. O_j receives the confirmation from O_i save the current state and sets its new state to X-bus.
11. O_i and O_j implement their I-code with the I-instructions to the shared and combined data parts from both sides.
12. Depending on the permissions O_i and O_j can read or write the data of each other.
13. The interaction of an object is terminated when its I-instructions are finished.
14. O_i and O_j go back to the checking phase.

5.2. *Mapping supporter MS*

In the previous section we have described the general architecture of the middleware KOM which supports KOs. In fact, the general model of KOM can support the mapping without

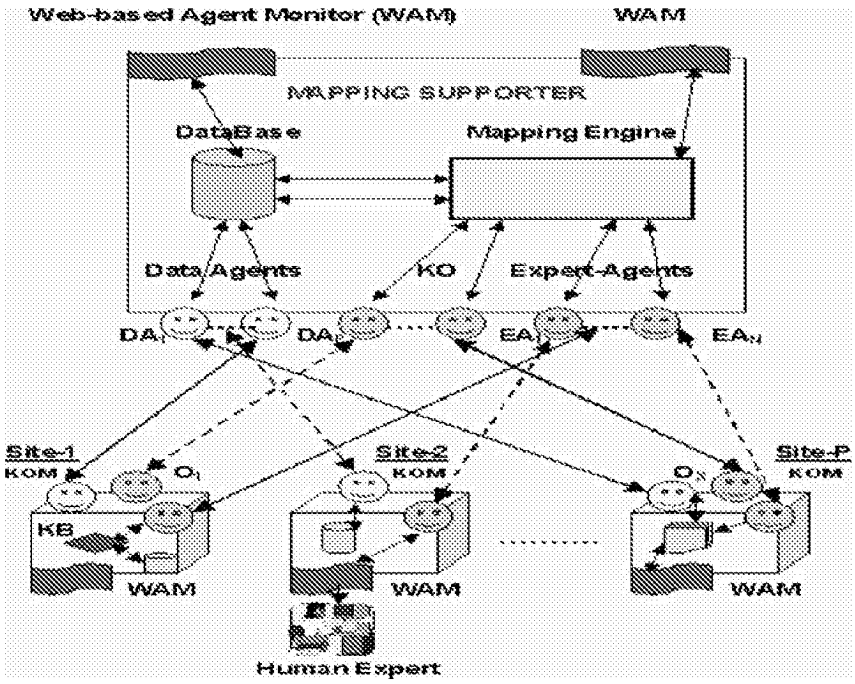


Figure 11. Agent-based mapping architecture.

any supporter. A study in [Ogston and Vassiliadis, 41] shows that this can be done via pure peer-to-peer interaction directly between agents which, in this case, are KOs. This method can provide relatively accurate solutions for one-criterion and one-value case. However, when pure peer-to-peer architecture is used, computation for a possible mapping O_i–O_j is done twice, one at the O_i site and one at the O_j site. Since the mapping with multiple criteria and multiple values requires more computation for each candidate pair O_i–O_j, the computation redundancy may overload the KOs. Therefore, we build a regional supporter for mapping MS to avoid this computation redundancy. The mapping supporter MS provides mapping-related services to KOs and is a part of the KO-services component of KOM. Thus, as a trade-off to deal with the bottleneck and the redundancy we divide the workload of mapping among the KOs and the MS. This is a combination of the peer-to-peer and the central computing models.

The mapping supporter MS is built based on agents as well. The arguments for using agents are that agents can carry out calculation competitively and in parallel, and can handle the distributed data and knowledge efficiently. The Internet agent-based architecture for mapping can be illustrated as in Figure 11. A mapping supporter MS may provide mapping services and information to a number of KOs. At each site the KOs run under the KOM's support. The interfaces of this system are Web-based (WAM), which allow users to modify the system easily at any location in case it is needed. For examples, users can create or delete KOs or agents, specify their locations or migration destination.

The mapping supporter MS consists of a Mapping Engine and a Mapping database (Figure 11) which are assisted by system agents. We have two kinds of system agents in this mapping system: (i) Data Agents, and (ii) Expert Agents, which may reside in the mapping supporter MS or at the business sites: site-1, site-2, ..., site-P. In addition to these agents the mapping supporter MS is open for interactions with KOs from the business sites.

The Data Agents are responsible for retrieving data about business environments such as evaluation criteria $\{C_1, C_2, \ldots, C_M\}$. Root Data Agents are created at the Mapping site and can be dispatched to the business sites. These traveler-agents can retrieve web-based documents of these sites locally and transfer data, which are already processed, to the Mapping Database via their root agents at the Mapping site. This is to reduce the network traffic.

Compared with the system in [Pham and Sharif, 46] the KOs $\{O_1, O_2, \ldots, O_X\}$ play the role of the Object Agents. They report their status to the Mapping Engine. In order to reduce network traffic in case MS and KOs are in different sites KOs may be able to create its child objects Oc_1 remotely at the mapping site, which acts as a representative of O_1 to provide up-to-date status of the given object to the Mapping Engine.

The Expert Agents act as the representatives of heterogeneous knowledge sources: $\{E_1, E_2, \ldots, E_N\}$ such as human specialists, intelligent software, knowledge-bases, to the Mapping Engine. Expert Agents receive tasks, which are to evaluate the business objects, from the Mapping Engine. They transform these evaluation requirements into queries to the target knowledge sources from $\{E_1, E_2, \ldots, E_N\}$. Expert agents are also responsible for providing necessary data to their knowledge sources using the Mapping database. The evaluation results are formed at the knowledge sources and then are transferred to the Mapping center through the Expert agents.

The Mapping Engine itself is an agent-based system which can be illustrated as in Figure 12. It defines the set of mapping criteria based on the information provided by Data Agents, and polls the Expert Agents to get the estimated business values $w[x, m, n]$ by each criterion C_m, $m = 1, \ldots, M$.

Requests for mapping come from the object interactions. They trigger the Mapping Generator which will call the Experts for participation and will form: (i) the data requirements, such as types and sources of data based on which the Data Agents are defined, and (ii) output requirements, such as which criteria will be used, which knowledge sources will be in the expert panel. Based on the mapping goals (local, global, or combined) provided by the Object Agents, the exact form of the mapping output will be defined.

Once the sources of data and knowledge, such as Data agents and $\{E_1, E_2, \ldots, E_N\}$, the scope of mapping such as $O = \{O_1, O_2, \ldots, O_X\}$, and the mapping requirements such as $\{C_1, C_2, \ldots, C_M\}$, are defined, the Mapping Kernel collects the evaluations from the committed experts by the defined criteria. These absolute business values $w[x, m, n]$, $x = 1, \ldots, X$, $m = 1, \ldots, M$, $n = 1, \ldots, N$, are kept updated by the Expert Agents and the objects. The Mapping Kernel includes a small database, a Data Transformer, a Fuzzy Ranker, and a Mapping Coordinator. The Data Transformer first converts absolute values $w[x, m, n]$ into the compatible values $v[x, m, n]$ using formula (1) and then into Interval-valued numbers by (2) as described in Section 2. Next, the Fuzzy Ranker defines the

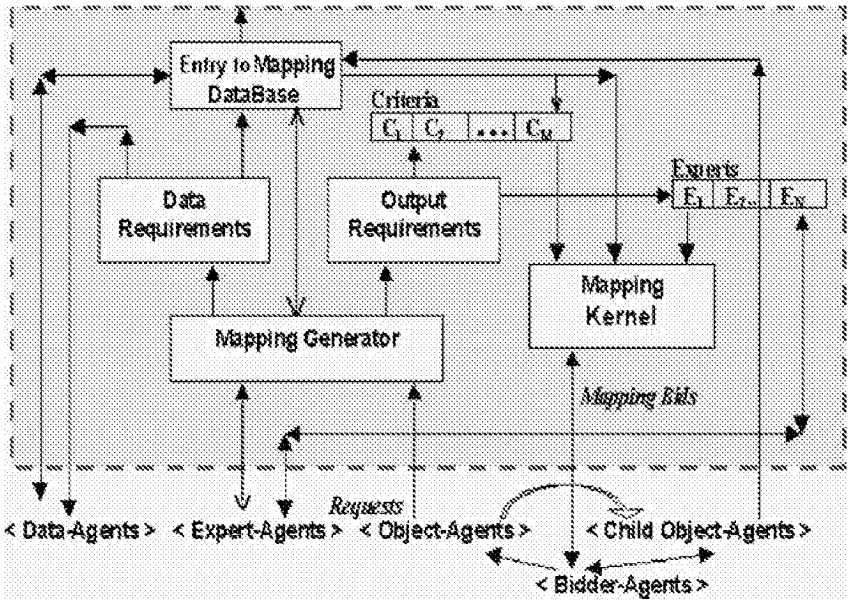


Figure 12. Structure of Mapping Engine.

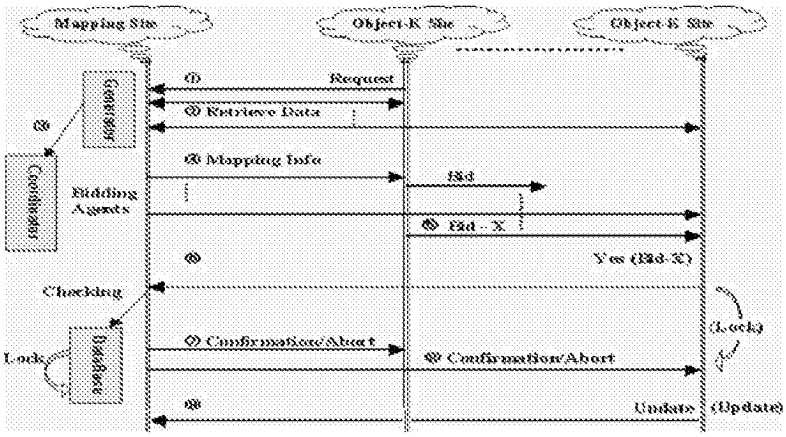


Figure 13. Mapping protocols by agent bidding.

mapping degrees $\mathrm{MD}_m(x, y)$ by (3) and then the combined degree $R(x, y)$ for each pair of objects.

Then, the Mapping Coordinator selects the most mapped pairs, by a given standard, for a competitive mapping which is implemented via bidder agents. The bidding protocols are described in Figure 13.

6. A case study: the e-city

To provide a heterogeneous, concurrent, and distributed environment for testing the proposed models we build an autonomous business system called "e-city". This is a complex virtual and digital business environment with different business scenarios and different distributions of data. Currently the e-city includes twelve different e-companies (Figure 14) and many e-citizens. Each e-company provides goods or services. They vary from e-bank, e-accounting, to e-school, e-entertainment, and e-mayor. Each e-citizen is managed by a graduate student, we have 30 of them, representing his/her needs or interests. Activities in the e-city involve the most popular issues of e-business: e-trading, e-management, e-learning, e-services, e-supply chains, and B2B. This e-business environment provides a various set of heterogeneous business demands and supplies which are represented via business objects. For examples, all of e-companies and e-citizens would generate e-business objects representing their different demands on e-accounts of various kinds. Meanwhile, the e-banks provide e-accounts and therefore would generate e-business objects representing their supply capacities for e-accounts. In total in the e-city we have a number of heterogeneous business objects carrying business demands and supplies which need to be matched into transactions by different criteria. The business logicstics and business knowledge in the e-city are stored in the KO-KnowledgeBase and can be changed and updated via the creation of objects which carry the changes and interact with the KnowledgeBase.

From the trading point of view, the e-city can be viewed as a set of complex transactions in nested, crossing, and overlapping dynamic supply chains. For example, the e-bank is the provider and the e-utility is its customer in term of money management. However, in term of material supply, the e-utility is the provider and the e-bank is its customer. Meanwhile, they are partners in sharing consuming and credit reports of the e-citizens. In order to

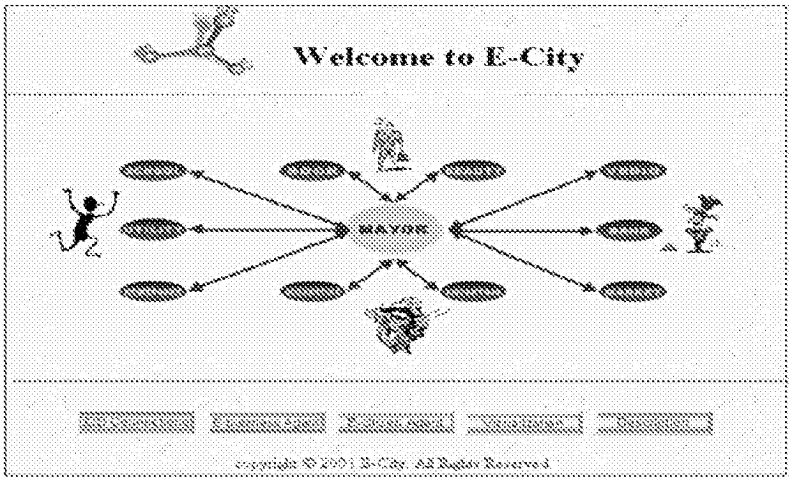


Figure 14. The autonomous business system "e-city"—a case study for ASIAM.

support the e-companies and e-citizens to conduct their businesses automatically we use B2B and B2C connectors under the Web interfaces. These connectors are agent-based and are activated at anytime when a transaction, as a result of mapped business objects, occurs. Thus, the E-city is a perfect environment and case study for the proposed agent-based mapping model.

7. Experimental results

In order to investigate how the developed methods in ASIAM with the interval-valued numbers and the knowledgeable objects as invisible agents can help to improve the performance and quality of service of E-business systems we conduct the following experiments on the interval-valued numbers and the agent invisibility.

7.1. *On the efficiency of interval-valued numbers*

We compare the efficiency and correctness in estimating dynamic business values of demands and supplies in using the Interval-based method and without it. In our experiments, $Q = 100$ business objects are generated, their business values $B(O_i, t)$, $i = 1, \ldots, Q$, are changed with a frequency of *Rbv* times/minute. The mapping degrees are calculated based on $B(O_i, t)$ in two ways: (i) without using IVN where we wait for the new value appears and then conduct the computation, and (ii) with IVN where the business value is calculated beforehand based on the past values and the recent trends. In order to measure these methods we use three criteria: (i) response time *Tres*—how long it takes to get the mapping degree from the moment when the new value appears, and (ii) the precision *Pre*—how accurate the obtained degrees are compared with what the current latest business values would produce. Notice that the business values are changing dynamically and it always take some time to calculate the mapping degrees no mater which mapping method is used. Therefore, if the business values are obtained at t and we get the mapping degrees at $t + d$ then those mapping degrees are actually for the t time moment not $t + d$ since business values may already have new values at the time moment $t + d$. The results on *Tres* and *Pre* are provided in Tables 1 and 2.

Table 1. Response time (average) on change rate.

Change rate\method	With IVN (PC time unit)	Without IVN (PC time unit)
0	7098	6997
10	7161	7040
20	7183	7155
30	7285	7180
40	7288	7299
50	7415	7366
60	7512	7468
70	7520	7546
80	7580	7589
90	7689	7658

Table 2. Precision (average) on change rate.

Change rate\method	With IVN (PC time unit)	Without IVN (PC time unit)
0	0.955	1.000
10	0.941	0.894
20	0.934	0.889
30	0.923	0.876
40	0.915	0.870
50	0.904	0.862
60	0.896	0.843
70	0.887	0.831
80	0.880	0.827
90	0.869	0.815

Table 3. Mapping efficiency on agent invisibility.

Visibility level Rvis (%)	Mapping time (K-units)	Number of mappings	% change of mapping time	% change of number of mappings
100	117,198	2,074	100.00	100.00
90	83,119	1,908	70.92	92.00
80	75,140	1,649	64.11	79.51
70	49,922	1,501	42.60	72.36
60	43,064	1,241	36.74	59.83
50	28,064	1,071	23.95	51.63
40	16,394	818	13.99	39.46
30	9,057	619	7.73	29.84
20	4,828	438	4.12	21.11
10	1,061	204	0.91	9.85

It shows that the model which uses IVN has almost the same response times compared with the one which does not use it, the difference is longer with 0.1–0.3%. However, the use of IVN provides more accurate results with 1–7% better.

7.2. *On the effectiveness of agent invisibility*

In order to investigate the influence of the invisibility of agents we conduct experiments with 10,000 agents by changing the invisibility level *Rvis* in the range 10–100% and measure how the mapping time and the number of mapping pairs would be changed. The results are shown in Table 3.

It shows that when the visibility level reduces, the number of mapping is also reduced. However, the mapping time decreases more substantially. In the large-scale systems if we keep 100% visibility, i.e. each object can "see" all the others then the mapping time is very long and could be unacceptable. Then, we can reduce the visibility until the mapping time satisfies the time constraints. For example, if we reduce the visibility to 70% the mapping numbers reduces 28% but the mapping time would reduce 43%!!! Thus, by regulating the invisibility of KOs we can make the mapping more flexible and adaptive to the changes of the requirements or to the system capacity. For example, when the density and the similarity of the demands and/or supplies are high we can decrease the visibility of KOs to decrease the mapping time without reducing the mapping quality substantially.

8. Related works and ASIAM

The general problem of mapping can have various forms and has applications in various fields such as process and memory management [Alain, 2], pattern recognition [Banzhaf, 6], image processing [Heckbert, 20], or database management [Rozen, 47], resource control in engineering and manufacturing [McMillan, 37], social study [Hamer and Cunningham, 19], data analysis [Smith, 51], supply chains [Bagchi et al., 5], automated trading [Dailianas et al., 12], or auctions [Kalagnanam et al., 26].

When the number of mapping criteria is more than one the mapping problem includes another intricate problem which is how to represent and combine many criteria within one single evaluation system, especially when the mapping criteria do not relate or are conflicting to each other. Multi-criterion mapping becomes extremely complicated when the mapping criteria cannot be evaluated by exact numbers because of the lack or the lost of information or because of the nature of the criterion. For examples, it would be difficult to represent preferences by numbers or it is impossible to predict the exact execution time of a business process in a dynamic system. These cases represent the most difficult form of the mapping problem when it involves uncertainty or incompleteness of data and has many criteria.

In order to solve the given problem we need to deal with several sub-problems:

- Represent incomplete data or the uncertainty.
- Evaluate criteria using representations of their incomplete data or the uncertainty.
- Map members of the set to each other based on the combined evaluation of all criteria.

Because of the complexity of the multi-criterion mapping problem with uncertainty and incomplete data, in practice, for such situations it usually requires human participation in making final decision. There are several results for each single sub-problem. However there have not been efficient solutions when the three sub-problems combine.

In the first sub-problem, incomplete data or uncertainty could be represented by probability [Hamer and Cunningham, 19; Zhang, 60], fuzzy numbers [Baas, 4; Dubois and Prade, 15], intervals [Yager and Kreinovich, 58; Nguyen et al., 40], or domains [Volgin and Levin, 56]. In the second sub-problem, the evaluation of a mapping by many criteria is carried out based on similarity measurement. There exit several similarity measurement methods which work with exact numbers: (i) correlation coefficients [McMillan, 37], (ii) distance measures [Everitt, 16], (iii) associate coefficients [Hamer and Cunningham, 19], (iv) probabilistic similarity coefficients [Clifford and Stephenson, 9]. In the third sub-problem, many mapping algorithms can be used. However, they are based on search methods with exact numbers only. For examples, Clustering [Meseguer et al., 38], Evaluation Function [Cong and Wu, 11], Genetic Algorithms [Keane, 29], Simulated Annealing [Kaddoura et al., 36], and Exhausted search [Herrmann, 21].

The proposed ASIAM differs from the other mapping mechanisms in the combination of using: (i) agents for representing and carrying data, (ii) interval-valued numbers, (iii) fuzzy ranking and comparison mechanisms for estimating business values, (iv) agent awareness and invisibility concepts, and (v) agent bidding for concurrent mapping.

In (i): the use of KO—a simple type of agents to represent each data item makes business data more active, dynamic to changes, and more interactive compared with the traditional ways of data representation such as files or databases, as explained in Section 2.

In (ii): the benefits of mapping using interval-valued numbers (IVN) compared with mapping using exact numbers [De and Roose, 14; Kaddoura et al., 36] are that an IVN accumulates past values and therefore can show more stable evaluations, if we use an exact number it would show only the value of a quantity at a time. Besides, IVNs also allow us to represent and to use the incomplete information such as "in between 5h and 7h" or "about 3". Finally, IVN with its multi-value capability is perfectly suitable to the mapping problem with multiple criteria and multiple experts.

In (iii): the modified fuzzy ranking mechanism allows ASIAM to rank quantities given in interval-valued numbers, while the traditional ranking mechanisms cannot.

In (iv): the agent invisibility concept allows us to dynamically regulate the scope of KO interactions and therefore helps to adjust the computation complexity, work load, and overheads of mapping KOs to the current system capacity in large-scale and distributed systems.

In (v): the differences of ASIAM compared with the mapping models without agents [Kaddoura et al., 36; Herrmann, 21; Irvin et al., 24; Keane, 29; De and Roose, 14] are that agents enable concurrent mapping and make the mapping results more responsive and more reflective to the dynamic changes in the e-business environment. Besides, the competitive mapping through agent bidders in ASIAM also reduces the complexity of the mapping process. Finally, agents make the mapping process more autonomous, mapping requirements and outputs are generated automatically without human intervention. This feature is essential in autonomous e-business processes and systems.

In summary, the proposed ASIAM is different from the other agent-based mapping systems such as [Huhns and Singh, 23; Dailianas et al., 12; Sycara et al., 55; Collins et al., 10; Gates and Nissen, 17] in using the interval-valued numbers and fuzzy ranking for representing and processing heterogeneous and related demands and supplies with multiple criteria. Besides, in ASIAM the KOs—agents are used to represent each single data item as a combined representation-processing tool with a one-to-one style while most other models build an agent for processing many data items, as a processing tool rather than as a representation tool and with a one-to-many style. While the approach of ASIAM makes data more active and self-processing it also leads to a side effect—a possible very high number of KO-agents which would require much more system resources compared with the other agent-based systems. That is why we propose the agent invisibility concept to decrease the scope of KO interactions when it is necessary, or in other words, to regulate the KO-agents activities and therefore to adjust the system work load in accord with the system capacity.

9. Conclusion

We have proposed an autonomous system, ASIAM, for dynamic mapping of demands and supplies with multiple criteria and multiple values in business Internet-based systems. The use of interval-valued numbers and fuzzy ranking for representing and clustering dynamic

business data allows us to include various kinds of business requests as demands and supplies, from selling and buying in trading to task assignments in management. In order to carry these heterogeneous demands and supplies in distributed business systems we developed light-weight agents with simple structure called knowledgeable objects. The agent awareness and invisible concepts are used to control the scopes of KO interaction and activities so that the workload dynamically meets the system capacity and the data distributions. Then, a distributed and autonomous mapping mechanism is implemented via agent biddings under the supports of system agents. The experimental results and our analysis shows that this combination of agent-based processing and interval-based representation not only meets the distributed, heterogeneous, and dynamic tendencies of e-business systems but also makes the mapping process more autonomous and effective.

This mapping system can be used in various e-business scenarios and systems, from inner organization management, e-marketplace, to supply chains, or inter organization management. Even though ASAIM is designed primarily for business systems it can also be applied to the mapping problem in other areas, such as dynamic control, scheduling, planning, pattern recognition, and other systems for data analysis and processing, where the data may be in different formats, locate in different locations, change dynamically, and may be incomplete.

The limitations of the current version of ASIAM are that while IVN can represent incomplete and conflicting data the system cannot handle conflicts at the agent level, it cannot recognize repeated demands and supplies, the interaction and the distance calculation overheads between KOs are high, and it would not be efficient when the demands or supplies are all substantially different from each other.

To improve the current system many further studies are needed such as: mapping algebra for reducing computation time, cooperation mechanisms for more efficient use of system agents, making the KOs—light-weight and invisible agents more reactive using the agent gender concepts in [Pham and Nguyen, 45], and more investigation on smart distance and agent awareness to reduce the distance calculation and increase the efficiency of invisibility concept.

Acknowledgments

This work is supported partially by the Research and Creative Projects Award RCA-00-02 from SUNY. Many thanks to all graduate students of the 25593/01 "E-business Computing" course for their participants in developing the E-city. Special thanks to the reviewers for their valuable comments.

References

[1] Abu-Hakima, S., C. McFarland, and J. Meech. (2001). "An Agent-Based System for Email Highlighting." In *Proceedings of Autonomous Agents*. ACM Press, pp. 224–225.

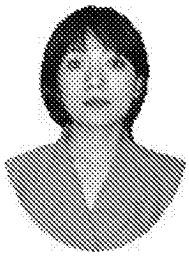
[2] Alain, D. (1993). "Mapping Uniform Loop Nests onto Distributed Memory Architectures." See `http://citeseer.nj.nec.com/.`

[3] Allen, J. (1983). "Maintaining Knowledge about Temporal Intervals." *Communications Magazine* 26, 832–843.

[4] Baas, K. (1977). "Rating and Ranking of Multiple Aspect of Alternatives Using Fuzzy Sets." *Automatica* 13(1), 47–58.
[5] Bagchi, S., S. Buckley, M. Ettl, and G. Lin. (1998). "Experience Using the IBM Supply Chain Simulator." In *Proceedings of the 1998 Winter Simulation Conference*. See `http://citeseer.nj.nec.com/`.
[6] Banzhaf, W. (1994). "Genotype-Phenotype-Mapping and Neutral Variation." See `http://citeseer.nj.nec.com/`.
[7] Bartelt, A. and L. Winfried. (2000). "Agent-Oriented Concepts to Forster the Automation of E-Business." In *Proceedings of the 11th International Workshop on Database and Expert Systems*, pp. 1–7.
[8] Bergamaschi, S. and D. Beneventano. (1999). "Integration of Information from Multiple Sources of Textual Data." In M. Klusch (ed.), *Intelligent Information Agents: Agent-Based Information Discovery and Management on the Internet*. Berlin: Springer, pp. 53–77.
[9] Clifford, H. and W. Stephenson. (1975). "An Introduction to Numerical Taxonomy." New York: Academic Press.
[10] Collins, J., C. Bilot, M. Gini, and B. Mobasher. (2001). "Decision Processes in Agent-Based Automated Contracting." *IEEE Internet Computing* 5(2), 61–72.
[11] Cong, J. and C. Wu. (1996). "An Improved Algorithm for Performance Optimal Technology Mapping with Retiming in LUT-Based FPGA Design." See `http://citeseer.nj.nec.com/`.
[12] Dailianas, A., J. Sairamesh, V. Gottemukkala, and A. Jhingran. (2000). "Profit-Driven Matching in E-Marketplace." In A. Moukas, C. Sierra and F. Ygge (eds.), *Agent Meditated Electronic Commerce (II)*. Lecture Notes in Artificial Intelligence, Vol. 1788. Berlin: Springer, pp. 153–179.
[13] Davis, E. (1990). "Representations of Commonsense Knowledge." San Mateo, CA: Morgan Kaufmann.
[14] De, J. and K.D. Roose. (1992). "Partitioning and Mapping Adaptive Multi-Grid Hierarchies on Distributed Memory Computers." See `http://citeseer.nj.nec.com/`.
[15] Dubois, D. and H. Prade. (1983). "Ranking Fuzzy Numbers in the Setting of Possibility Theory." *Information Sciences* 30, 183–224.
[16] Everitt, B. (1980). "Cluster Analysis." Halsted Publisher.
[17] Gates, W.R. and M.E. Nissen. (2001). "Designing Agent-Based Electronic Employment Markets." *Electronic Commerce Research* 1(3), 239–263.
[18] Geffner, H. and B. Bonet. (1998). "High-Level Planning and Control with Incomplete Information Using POMDPs." In *Proceedings of the Workshop on IPSEDUE, AIPS'98*, pp. 113–120.
[19] Hamer, R. and Cunningham. (1981). "Cluster Analyzing Profile Data Confounded with Iterate Differences: A Comparison of Profile Association Measures." See `http://citeseer.nj.nec.com/`.
[20] Heckbert, P.S. (1986). "Survey of Texture Mapping." See `http://citeseer.nj.nec.com/`.
[21] Herrmann, C. (1996). "On the Space–Time Mapping of a Class of Divide-and-Conquer Recursions." See `http://citeseer.nj.nec.com/`.
[22] Horn, P. (2001). "Autonomic Computing: IBM's Perspective on the State of the Information Technology." *IBM Research*, 1–22.
[23] Huhns, M. and M. Singh. (1998). "Managing Heterogeneous Transaction Workflows with Co-Operating Agents." In N. Jennings and M. Wooldridge (eds.), *Agent Technology*. Berlin: Springer, pp. 219–240.
[24] Irvin, R.B. et al. (1995). "Mapping Performance Data for High-Level and Data Views of Parallel Program Performance." See `http://citeseer.nj.nec.com/`.
[25] Jensen, D., Y. Dong, B. Lerner, E. McCall, L. Osterweil, S. Sutton, Jr., and A. Wise. (1999). "Coordinating Agent Activities in Knowledge Discovery Processes." In *Proc. of Work Activities Coordination and Collaboration Conference*. See `http://citeseer.nj.nec.com/`.
[26] Kalagnanam, J.R., A.J. Davenport, and H.S. Lee. (2001). "Computational Aspects of Clearing Continuous Call Double Auctions with Assignment Constraints and Invisible Demand." *Electronic Commerce Research* 1(3), 221–238.
[27] Karacapilidis, N. and P. Moraitis. (2001). "Intelligent Agents for an Artificial Market System." In *Proceedings of Autonomous Agents*. ACM Press 2001, pp. 592–599.
[28] Kargupta, H., I. Hamzaoglu, and B. Stafford. (1997). "Scalable, Distributed Data Mining—An Agent Architecture." In *Proceedings of the 3rd International Conference on Knowledge Discovery & Data Mining*. AAAI Press. See `http://citeseer.nj.nec.com/`.

[29] Keane, M. (1994). "Adaptation as a Selection Constraint on Analogical Mapping." See http://citeseer.nj.nec.com/.
[30] Koenig, S. and R.G. Simmons. (1998). "Solving Robot Navigation Problems with Initial Pose Uncertainty Using Real-Time Heuristic Search." In *Proceedings of the 4th AIPS*. See http://citeseer.nj.nec.com/.
[31] Kraus, S. and D. Lehmann. (1995). "Designing and Building an Automated Negotiation Agent." *Computational Intelligence* 11(1), 132–171.
[32] Kreinovich, V. (1995). "Data Processing Beyond Traditional Statistics: Applications of Interval Computations: A Brief Introduction." *Journal of Reliable Computing*. See http://citeseer.nj.nec.com/.
[33] Levin, V.I. and R. Trejo. (1998). "On Comparison of Interval Quantities." Technical Report, Penza Technological University. See http://citeseer.nj.nec.com/.
[34] Lieberman, H., B. Nardi, and D. Wright. (2001). "Training Agents to Recognize Text by Example." *Journal of Autonomous Agents and Multi-Agent Systems* 4(1/2), 79–92.
[35] Maes, P., R. Guttman, and A. Moukas. (1999). "Agents that Buy and Sell: Transforming Commerce as We Know it." *Communications Magazine* 42, 81–91.
[36] Kaddoura, M., C.-W. Ou and S. Ranka. (1995). "Mapping Unstructured Computational Graphs for Adaptive and Non-uniform Computational Environments." *IEEE Parallel and Distributed Technology* 3(3), 63–69.
[37] McMillan, L. (1992). "A Forward-Mapping Realization of the Inverse Discrete Cosine Transform." See http://citeseer.nj.nec.com/.
[38] Meseguer, J. et al. (1998). "Mapping Tile Logic into Rewriting Logic." See http://citeseer.nj.nec.com/.
[39] Minar, N. (2001). *A Network of Peers, Peer-To-Peer: Harnessing the Power of Disruptive Technologies*, A. Oram (ed.). O'Reilly Publisher.
[40] Nguyen, T. Hung, V. Kreinovich, and Zuo Qiang. (1997). "Interval Valued Degrees of Belief: Application of Interval Computations to Expert Systems and Intelligent Control." *Int J. of Uncertainty, Fuzziness and Knowledge-Based Systems* 5(3), 317–358.
[41] Ogston, E. and S. Vassiliadis. (2001). "Matchmaking Among Agents without a Facilitator." In *Proceedings of Autonomous Agents*, pp. 608–615.
[42] Pham, H. (1999). "Fuzzy Estimation for Interval-Valued Numbers with Statistical Information." In *Proceedings of ISFL'99*, pp. 62–68.
[43] Pham, H. (2001). "Knowledgeable Objects for Data Representation and Data Processing." Technical Report, SUNY New Paltz, CS-HP01-07-01, pp. 1–30.
[44] Pham, H. (2001). "Software Agents for Internet-Based Systems and Their Design." In J. Lakhmi and C. Zhengxin (eds.), *Intelligent Agents for System Design*. Physica-Verlag, pp. 109–155.
[45] Pham, H. and V.H. Nguyen. (2001). "Agents with Genders for Inventory Planning in E-Management." In E. Stroulia and S. Matwin (eds.), Lecture Notes in Computer Science, Vol. 2056. Springer, pp. 267–277.
[46] Pham, H. and S. Sharif. (2001). "Competitive Mapping Business Objects with Interval-Valued Numbers." In *Proceedings of the International Conference on Artificial Intelligence (IC-AI'2001)*, Vol. III. CSREA Press, pp. 1178–1185.
[47] Rozen, S. (1995). "LabBase: A Database to Manage Laboratory Data in a Large-Scale." See http://citeseer.nj.nec.com/.
[48] Sandhom, T. (1999). "Automated Decision Making." In G. Weis, W. Hoek, and Wooldrige (eds.), *EASSS*. Cambridge, MA: MIT Press, pp. 201–258.
[49] Sen, S., P. Dutta, and R. Mukherjee. (2000). "Agents that Represent Buyer's Interests in E-Commerce." See http://citeseer.nj.nec.com/.
[50] Sheth, A. (1999). "Workflow Automation: Applications, Technology, and Research." In *Proceedings of SIGMOD Conference*.
[51] Smith, W.D. (1994). "Accurate Circle Configurations and Numerical Conformal Mapping." See http://citeseer.nj.nec.com/.
[52] Somlo, G. and A. Howe. (2001). "Incremental Clustering for Profile Maintenance in Information Gathering Web Agents." In *Proceedings of Autonomous Agents*. ACM Press, pp. 262–269.

[53] Stone, P., M. Littman, S. Singh, and M. Kearns. (2001). "ATTac-2000: An Adaptive Autonomous Bidding Agent." In *Proceedings of Autonomous Agents*. ACM Press, pp. 238–245.
[54] Strobel, M. (2001). "Design of Roles and Protocols for Electronic Negotiations." *Electronic Commerce Research* 1(3), 335–353.
[55] Sycara, K., M. Klusch, S. Widoff, and J. Lu. (2001). "LARKS: Dynamic Matchmaking among Heterogeneous Software Agents in Cyberspace." *Journal of Autonomous Agents and Multi-Agent Systems*.
[56] Volgin, L.I. and V.I. Levin. (1991). *Continuous Logic: Theory and Applications*. Estonian Academy of Sciences Press.
[57] Walsh, W. and M. Wellman. (2000). "Modeling Supply Chain Formation in Multi-Agent Systems." In A. Moukas, C. Sierra, and F. Ygge (eds.), *Agent-Mediated E-Commerce II*, Lecture Notes in Computer Science, Vol. 1788. Berlin: Springer, pp. 94–101.
[58] Yager, R. and V. Kreinovich. (1998). "Decision Making under Interval Probabilities." Technical Report UTEP-CS-98-12.
[59] Ye, Y., S. Boies, P. Huang, and J. Tsotsos. (2001). "Smart Distance and WWWaware—A Multi-Agent Approach." In *Proceedings of Autonomous Agents*. ACM Press, pp. 176–177.
[60] Zhang, W. (1998). "Flexible and Approximate Computation through State-Space Reduction." In *Uncertainty in Artificial Intelligent '98*. San Mateo, CA: Morgan Kaufmann, pp. 531–538.



Hanh Pham is an Assistant Professor at the Department of Computer Science, State University of New York at New Paltz. She received a Ph.D. in computer science from National Technical University of Ukraine. Her research interests include autonomic computing, autonomous control, distributed systems, heterogeneous data processing, fuzzy logics, and multi-agent systems.

Yiming Ye is a Research Staff Member at the Department of Electronic Commerce Research, IBM TJ Watson Research Center. He received his Ph.D. degree in computer vision from University of Toronto, Canada. His current research interests are real-time business collaboration systems, agent and multi-agent systems, and applied dynamics of complex systems.

Vien Nguyen is an employee of a top-tier financial firm. Previously she was an Associate Professor at the Sloan School of Management at MIT. She holds a Ph.D. in operations research from Stanford University and an A.B. in Applied Mathematics from Harvard College.