Abstract: This paper presents a grid architecture for E-Service. It comprises nine semantic components: port, probe, adapter, action, activity, workflow, E-Service, constraint and trigger, which form four different integration levels while each level could handle the semantics of presentation, function and resource separately. This grid architecture may facilitate us implementing E-Commerce automation and semantic based composition. We mainly focus on the possible structure of nine semantic components and the composition based on the grid architecture in this paper.

Key-Words: E-Commerce automation, Architecture, Semantics, Workflow, Integration, Composition.

1 Introduction
E-Commerce ultimately boils down to E-Service automation that affords a platform for automating both the provider and consumer ends business transactions. “In spite of the significant potential benefits, only a few large business have implemented automated E-Business so far” [1].

The cost of automating transaction with trading partner is very high. One important costs-down-drive is a flexible software architecture, which “is much like a city plan in that it defines a blueprint that will meet the current and future needs of a diverse user population, and will adapt to changing business and technology requirements” [2].

E-Commerce platform is a “composition” of business strategies and technologies [3]; both of them are changing frequently. Therefore, introducing a middleware to cut off the direct coupling between business strategies and technologies would facilitate businessman and researcher/developer concentrating on their own specialty as well as bridging the gaps between them. We call this middleware as elementary E-Service layer. In our opinion, each elementary E-Service should focus on one single function. It is building block of E-Commerce platform while business strategies are guidelines of how to synthesize elementary E-Service.

This paper presents a grid architecture for E-Service. It has four different integration levels while each level is divided into three parts, handling the semantics of presentation, function and resource respectively. It could be used to fulfill semantic based composition and E-Commerce automation. In the remainder of this paper, related work on architecture is summarized in section 2. The function and possible structure of nine semantic components are discussed in section 3. The overall view of the grid architecture and its composition scenario are illustrated in section 4. Section 5 concludes this paper.
2  Related Work

E-Commerce architecture design issues drew great attentions even at the very beginning of this booming tendency. Client-server architecture and three-tiered (multi-tiered) architecture is the first stage. These architectures intend to segment E-Commerce platform into distinct layers. Although different layers are loosely coupled, business strategies and technologies are still tightly coupled internally in the application service [2].

To keep E-Commerce platform grounded in reality, it’s best to take an incremental approach. Many researchers found that there should be several levels to integrate a successful E-Commerce platform [4, 5]. It indicates that the business logical layer also demands a flexible architecture for the current and future integration, cooperation and composition. One solution is object-oriented architecture, such as Eco system [6]. However, components/objects are very small reusable units for developers to construct an E-Commerce platform and the interoperability issues between them are always being ignored.

Some components/objects that are bound together to perform certain tasks could also be employed as “plug and play software devices”. These forms process-based, agent-based, transaction-based or workflow-based integration, such as AB-OOBMC [7], process based inter-enterprise integration [8], architecture-centric approach for multi-agents system [9], InterMarket [10], GAIA [11], MST based workflow interoperation [12], multi-level schema based workflow [13], etc.

In fact, agent is “a process that lives in the world of computers and networks that can operate autonomously to fulfill a set of tasks” [14], workflow is “concerned with the automation of procedures where information and tasks are passed between participants according to a defined set of rules to achieve, or contribute to, an overall business goal” [15] and transaction could be regard as a procedure characterized by ACID properties. Therefore, the essential of these kinds of integrations is identical; they only emphasize different sides of the problems. Now, all these researches are still ongoing; their common shortcoming is that they must predefine all the potentially reachable situations [2, 12].

Very recently, semantic based composition of E-Commerce has attracted some attentions [3, 16]. In this paradigm, composition of services needs is generated on the fly based on the requests of the customers and could be modeled not requiring programming at all. Because this concept is still in its infancy, no architecture issues about it have been discussed yet. This paper focuses on this topic.

3  Architecture Components

The grid architecture comprises several semantic components, including port, probe, action, activity, workflow, E-Service, adapter, trigger and constraint. Among them, action, activity, workflow and E-Service are core components that form four different integration levels while the other components attach themselves to different core components and/or present at different integration level to construct an E-service.

These semantic components are described in the following sections.

3.1  Port and Probe

Action, activity, workflow and E-Service use ports to exchange parameters, messages, events, contracts, handles of presentation entities, resource links and other formal information between each other [17]. Besides the formal information that would be encapsulated by XML doc, port possesses its own constraints and triggers that implement the operation rules acted on it.

IN ports should be distinguished from OUT ports. IN ports are used to pass formal information into a core component while OUT ports make formal information of a core component available to the outside.

Ports also define the boundaries of action, activity, workflow and E-Service. These boundaries could be used to carry out projection, which means to get parts of the core components.

Port has the following structure:

\[
\text{Port} = (C, FI, T, IN|OUT)
\]

Where \( C = \text{Constraints} \), \( FI = \text{Formal Information} \), \( T = \text{Triggers} \) (1)

Probes are special OUT ports, which expose the runtime status of a core component. However, they cannot be used to define the boundaries of a core component and cannot be used as delimiters of projection.

Of course, how to deploy probes depends on the agreements formally established between the requester and provider of the components. Therefore, probe has some different structure with port:

\[
\text{Probe} = (C, FI, T, L)
\]

Where \( C = \text{Constraints} \), \( FI = \text{Formal Information} \), \( T = \text{Triggers} \), \( L = \text{Locations} \) (2)

The information exposed by ports and/or probes could be employed to implement visualization, audit, monitor, access control, etc.
3.2 Adapter
The formal information passed between IN ports and OUT ports (probes) may have different formats. The control model between two interacting core components could also create “gaps”, e.g., a component that uses step-by-step or batch control may not be able to respond to requests coming from a component using an event-driven control model [20]. Adapters can be used to bridge these gaps.

Adapter is also called transformation [17]. One important thing is that the transformation must preserve the origin semantics in addition to syntax transformation. Adapter could be defined as:

\[
\text{Adapter} = (I, PL, PB, O)
\]

Where \( I = \text{IN Ports} \), \( PL = \text{Transformation Policies} \), \( PB = \text{Probes} \), \( O = \text{OUT Ports} \) (3)

Because there may be numbers of ports linked to one adapter, it could also be regard as an event channel [21].

3.3 Action
Action is a semantic wrapper of indivisible procedure or function, i.e., it adds the semantics of presentation, function and resource to the traditional procedure or function.

Procedure and function are reusable programming units for developer. To enhance the feasibility of integration, some standard description file had been used to wrap procedures and functions, such as CORBA IDL file. Now, we are going to wrap them with some XML based doc. The concept support this is our semantic action.

Each action has just one IN port and one OUT port. Meanwhile, it may have several probes. The structure of action is:

\[
\text{Action} = (I, SP, SF, SR, PB, O)
\]

Where \( I = \text{IN Port} \), \( SP = \text{Semantics of Presentation} \), \( SF = \text{Semantics of Function} \), \( SR = \text{Semantics of Resource} \), \( PB = \text{Probes} \), \( O = \text{OUT Port} \) (4)

Because the procedure or function wrapped in action is indivisible, projection on an action is invalid.

3.4 Activity
Similarly, activity is a semantic wrapper of traditional transaction, a procedure characterized by ACID properties.

Each activity must possess two OUT ports; one is used to commit the wrapped transaction while the other one is used to abort, cancel or rollback it. In other aspects, activity and action are almost identical. The structure of activity is:

\[
\text{Activity} = (I, SP, SF, SR, PB, O)
\]

Where \( I = \text{IN Port} \), \( SP = \text{Semantics of Presentation} \), \( SF = \text{Semantics of Function} \), \( SR = \text{Semantics of Resource} \), \( PB = \text{Probes} \), \( O = \text{Two OUT Ports} \) (5)

Of course, projection on an activity is also invalid.

3.5 Workflow
Traditional workflow encodes actions, activities and the relationships among them. It can be graphically depicted with nodes denoting actions and/or activities and arrows denoting precedence. Typically, there are four control relationships among actions and activities: OR-split, AND-split, OR-join and AND-join. The former two relationships are used to specify branching decisions in a workflow while the latter two specify points where actions and/or activities converging [19].

In the grid architecture, workflow is a semantic wrapper of traditional workflow. It may have numbers of IN ports, OUT ports and probes. Meanwhile, it has a presentation semantic, a function semantic and a resource semantic respectively. The presentation semantic is a visualization view of workflow; it provides some GUI to facilitate the users, such as webflow [13]. The resource semantic is a resource view of workflow; it indicates the underlying procedures of resources requisition. The function semantic indicates the tasks involved and their navigation relationships. The semantic structure of workflow is:

\[
\text{Workflow} = (I, SP, SF, SR, PB, O)
\]

Where \( I = \text{IN Ports} \), \( SP = \text{Semantics of Presentation} \), \( SF = \text{Semantics of Function} \), \( SR = \text{Semantics of Resource} \), \( PB = \text{Probes} \), \( O = \text{OUT Ports} \) (6)

With this structure, the projection, integration and outsourcing of workflows involve two phases.
1. Projection, integration and outsourcing on workflow semantics.
2. Projection, integration and outsourcing on underlying presentation, function and/or resource views. One scheme has been discussed in [19].

In our opinion, workflows are the concrete implementations of E-service.
3.6 E-Service
Even at the very beginning, it was noticed that we should represent E-Service as a semantic model [3]. E-Service refers to self-contained, internet-enable applications capable not only of performing business activities on their own, but also possessing the ability to engage other E-Service in order to complete higher-order business transactions.

Main characteristics that make E-Service composition very different from workflow integration and software component integration are summarized in [3]. It is rather polymorphous, e.g., it can be data oriented, process oriented, transactional or combinations of all these kinds.

Main problems of E-Service composition are how to describe E-Service, how to advertise and discover E-Service, how to perform E-Service matching and compatibility checking, how to implement outsourcing scheme, how to synthesize E-Service, etc. [3, 16, 19, 22]. All these problems should be handled by using semantic model of E-Service. In the grid architecture, E-Service could be represented by the following structure:

Service = (I, SP, SF, SR, PB, O)
Where I = IN Ports,
SP = Semantics of Presentation,
SF = Semantics of Function,
SR = Semantics of Resource,
PB = Probes,
O = OUT Ports
(7)

On the other hand, E-Service composition could be achieved by exchange quotation [23], XML messages [12], contract [22], profile [16] or PO/POA [17] between each other. In fact, all of them are XML based semantic information.

E-Service is a published interface of underlying workflows; it describes itself as profiles.

3.7 Constraint and Trigger
Business strategies could be mapped into integrity constraints and/or trigger rules [24]. In the grid architecture, constraint and trigger are also described as semantic models.

Because ports are delimiters of action, activity, workflow and E-Service, we suppose constraint and trigger could be applied to these core components only by linking themselves to relevant ports or probes. Therefore, ports and probes are the entries of constraints and triggers.

The same business strategy may be expressed in different manner for different core components. For example, at the E-Service level, it would be “orders greater than 500 units receive a 30% discount”; at the workflow level, it would be “when orders greater than 500 units, the activity of discount must be executed with the parameter of 30%”; at the activity level, it would be “when the threshold value ranges from 500 to infinity, discount ratio equals 30%”. We can find two important facts from above example:

1. Constraints and triggers have their own scopes. A constraint or trigger at E-Service level would affect all actions, activities and workflows involved while a constraint or trigger at action level would just affect the action.
2. Constraints and triggers would be derived from constraints and triggers at the upper integration level. So, we must pay attention to their sources and relationships.

\[\text{Constraint} \mid \text{Trigger} = (I, SR, PB, O)\]
Where I = IN Ports,
SR = Semantics of Rules
PB = Probes
O = OUT Ports
(8, 9)

Of course, besides the rule expressions, other information, such as scope, parent, etc., must also be wrapped in SR description.

The class diagram of the architecture components is shown in Fig. 1. All associations between the components could be summed up as “possess”.

4 The Grid Architecture
The core architecture components of action, activity, workflow and E-Service form four different integration levels for E-Commerce platform. These levels divide the business logical layer into four sub-layers. On the hand, because each sub-layer has its own presentation, function and resource semantics, the overall grid architecture is shown in Fig. 2.
The grid architecture for E-Service composition allows for the seamless integration of various components, including XML, HTML, CORBA, and DCOM, to facilitate efficient communication and data exchange.

![The Grid Architecture for E-Service](image)

**Fig. 2 The Grid Architecture for E-Service**

The scenario of E-Service composition is shown in Fig. 3. Company A builds its E-Service using the grid architecture and stores them in the local E-Service Repository (step 1). Company A advertises its E-Service profiles to the E-Service Registry (step 2). Company B submits its E-Service requests to E-Service Registry. After performing profiles matching and compatibility checking, E-Service Registry returns the composition plan to company B, including composition alternatives. Different E-Service requesters and providers (company A and B) must achieve an agreement before any composition (step 6). After that, composition could be carried out simultaneously at different sites (step 7). If needed, some new E-Service must also be registered to E-Service Registry (step 8). Afterwards, company A and B could do business electronically.

![The Scenario of E-Service Composition](image)

**Fig. 3 The Scenario of E-Service Composition**

To support the scenario described above, we need the following main technologies to develop a complete solutions for E-Service composition: semantic specification can be used to describe all kinds of architecture components, description of composition request and composition plan, matching and compatibility checking scheme, projection and union algorithms, etc. Some of them had been discussed in [3, 19]. For the grid architecture, two other important issues must be noticed:

1. Semantic interpreter used to translate the semantic specifications between two adjoined integration levels.
2. Semantic mediator used to merge incompatible constraints and triggers at the same integration level.

Of course, these topics are still under research.

### 5 Conclusion

Nine semantic components, including port, probe, adapter, action, activity, workflow, E-Service, constraint and trigger form four different integration levels to build E-Commerce platform. Each level could handle the semantics of presentation, function and resource separately. It shapes into a grid architecture that may help us to implement E-Commerce automation and semantic based composition.

Because these technologies are still in their infancy, a lot of works must be done to implement, improve and justify the grid architecture.

### References:


