RC23200 (W0405-003) May 3, 2004
Computer Science

IBM Research Report

Unified Commerce Server Architecture for Large Number of Enterprise Stores

Trieu C. Chieu, Florian Pinel, Jih-Shyr Yih
IBM Research Division
Thomas J. Watson Research Center
P.O. Box 704
Yorktown Heights, NY 10598

Research Division
Almaden - Austin - Beijing - Haifa - India - T. J. Watson - Tokyo - Zurich

Unified Commerce Server Architecture for Large Number of Enterprise Stores

Trieu C. Chieu, Florian Pinel, Jih-Shyr Yih
IBM T. J. Watson Research Center
19 Skyline Drive, Hawthorne, NY 10532, USA
e-mail: {tchieu, pinel, jyih}@us.ibm.com

ABSTRACT

To serve a large number of enterprise customers, enterprise supplier needs to set up many custom B2B stores, each personalized with special promotions and dedicated contracts. The custom store needs to empower business buyers to shop in the following fashions: 1) Buyers visit the custom store, make direct decisions on product selection, based on entitled pricing, and check out orders; 2) Buyers use their own company's procurement software, perform catalog shopping on one or more remote custom stores, return an order quote back locally for approval, and subsequently send approved orders to supplier for fulfillment; 3) Buyers shop with own company's procurement software, using catalog pre-loaded with product and pricing information extracted from supplier's custom stores, and then place purchase orders. This paper describes an industry implementation experience, on how these shopping paradigms can be accommodated effectively and efficiently using a unified commerce server architecture. It is explained, how one master catalog subsystem can be personalized to accommodate the following business processes: B2B Direct catalog shopping, B2B procurement with remote catalog punchout, B2B procurement with local catalog, and entitled catalog extraction. A detailed discussion on how to build a unified commerce server in a 2-tier architecture is illustrated by the handling of various different punchout protocols in one common implementation. The study also reports on the issues and efficiency considerations by different approaches for real-time entitled pricing lookup vs. batch pricing extraction operations.

Keywords

Electronic Commerce, Business-to-Business, Procurement, Punchout, Catalog, Entitled Price

1. INTRODUCTION

To strengthen customer service operations, streamline supply chains, and reach existing and new customers, companies have been enabling their core business with online commerce. Leading companies are drawing on powerful business-to-business (B2B) applications to exchange information, and interact and execute business transactions with customers and suppliers throughout their value chains. The earliest applications relied on a point-to-point connection for exchanging data electronically, such as electronic data interchange (EDI)[1], which is difficult and costly to implement, and requires a unique solution for each pair of trading partners. As the Internet technology prevails, Web-based EDI, commerce portals and B2B sites become mainstream to enable Web-based business interactions and purchase transactions with up-to-date real-time information, richer content, easier navigation and better user experience.

To serve a large number of enterprise customers, supplier companies need to set up many custom B2B stores, each personalized with special promotions and dedicated contracts. Some customers choose to shop directly in their dedicated B2B stores, others may want to use commercial procurement solutions to conduct purchases. The complexity of implementing and supporting consistent contents and pricings in an enterprise commerce engine for both direct shopping or indirect purchases from various commercial procurement systems has burdened many suppliers with increasing administrative and maintenance costs. It is therefore desirable to introduce a more adequate and efficient system solution to handle this increasingly complex problem.

In this paper, we introduce a unified commerce server in a two-tier architecture to support a large number of custom B2B stores for both direct catalog shopping and purchases through procurement integration. We describe how the unified server with a single master catalog can be personalized to accommodate various shopping paradigms including direct catalog shopping, B2B procurement with remote catalog punchout or local catalog[2, 3]. The paper is structured as follows. In Section 2, we first present a two-tier commerce architecture with an enterprise B2B gateway and a unified commerce server to support both B2B interaction between external and internal systems, and B2B catalog shopping through portal access. We then describe the architecture objective and strategic position of the unified commerce server named as Common Commerce Engine. In Section 3, we explain how a single master catalog in a common engine can be designed to serve multiple custom B2B stores. In Sections 4, we present procurement solutions and discuss how to effectively handle various commercial procurement systems with our unified commerce server. We also present and discuss the issues and efficiency considerations on procurement integration by different approaches with respect to real-time entitled pricing lookup vs. batch pricing extraction operations. Finally, summary and conclusions are given in Section 5.

2. UNIFIED COMMERCE SERVER ARCHITECTURE

Enterprise suppliers such as IBM offer a variety of custom B2B stores to allow online purchasing of a variety of products/supplies for their large enterprise customers, affiliates, and business partners. In the past years, different divisions in IBM have implemented and provided similar functions aimed at particular audience needs, each has evolved separately and uses differing variations of underlying technology and middleware. As part of IBM e-business strategy, an unified commerce server with a two-tier architecture has developed to leverage IBM commerce technology and infrastructure to provide a high degree of ability for rapid change, and insure flexibility to support multiple audiences through common, modular components. In the next sub-sections, we present the details of this architecture and its constituent systems.

2.1 Two-Tier Systems for Separation of Responsibilities

The 2-tier architecture, as depicted in Figure 1, is designed to decouple the tasks involved for B2B interactions into separate responsibilities and assign the sub-tasks to separate responsible systems. The benefits gained are scalability and flexibility. In this respect, we implement an enterprise B2B gateway system in the first tier that is positioned to support all internal business units that require B2B interfaces to interact with external systems. In the second tier, we implement an unified commerce server with a single code base that consolidates multiple commerce engines currently in use in IBM into a single hosting environment to serve the various custom B2B stores for enterprise customers.

In general, there are many entry points to access the B2B stores hosted in the commerce engine. Some possible entry points, as illustrated in Figure 1, are: 1) business partners entering from business partner portals; 2) affinity groups such as stock holders, or employees, accessing their corresponding portals; 3) large enterprise customers entering from secure extranet-site (e-Site) portals; and 4) procurement systems connecting to B2B gateway via a commercial eProcurement network.

In cases 1), 2) and 3) above, the customers can access their custom B2B stores using a browser through dedicated portal URLs. After authentication to a portal, the enterprise customers will be presented with a list of pre-authorized B2B sites with entitled products for purchases. In case 4) above, the enterprise customers use their procurement applications to link to a B2B gateway and custom B2B stores via some commercial procurement networks for shopping and ordering, and to request for a return of the contents of the shopping cart as an order quote back to their procurement applications for approval. In this later case, the procurement customers are also presented with the same entitled catalogs for shopping as they are accessing the custom B2B stores directly through a browser interface.

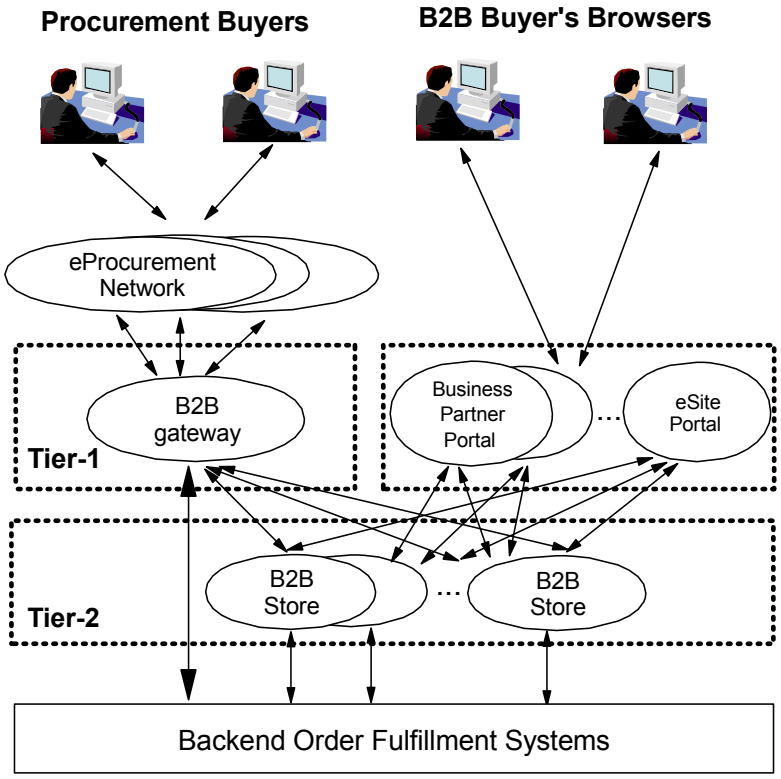

Figure 1: Two-tier architecture connected to backend order fulfillment systems

In the case of procurement integration via an eProcurement network using commercial systems, there are further requirements that the system needs to handle various procurement protocols. Investigation shows that the integration process can be decoupled into two protocol-dependent and protocol-independent tasks. The protocol-dependent tasks are related to those required for interfacing with external eProcurement networks, while the protocol-independent tasks are the core commerce functions that have to be provided independently of the requesting protocol and

access point. It is obvious to that the protocol-dependent tasks should be assigned to the fist-tier B2B gateway system, while the protocol-independent tasks should be assigned to the second-tier commerce server.

2.2 Enterprise B2B Gateway

The IBM Enterprise B2B Gateway, which is referred as the tier-1 enterprise B2B gateway in our solution, is a gateway system implementing B2B protocol interfaces to perform extensive set of business collaborative interactions between business partners and IBM backend order and fulfillment systems. It is positioned as a shared edge of enterprise solution to support all IBM business units with systems that require B2B interfaces. It primarily supports server-to-server interfaces, between IBM internal systems and IBM's trading partners, over multiple external protocol formats, such as RosettaNet[4], Open Application Group's (OAG) Business Object Documents[5], EDI interfaces, proprietary formats like Ariba cXML, mySAP Open Catalog Interface (OCI) [6] and custom formats, File Transfer Protocol and Lotus Notes database replication.

Both buy-side and sell-side processes are supported in IBM B2B Enterprise Gateway. Examples of key buy-side processes including submitting purchase orders to suppliers, receiving order status, invoice/remittance, and sending and receiving collaborative messages such as *advance shipping notices, forecasts and forecast commits*. Examples of key sell-side processes[7] include integration support for external procurement systems, receiving purchase orders from customers and business partners, providing orders and shipping status, invoice, and entitled catalogs, and sending inventory data.

For procurement integration support, the gateway implements a number of B2B connectors that are used to interface with commercial eProcurement packages such as Ariba Commerce Service Network, CommerceOne, mySAP. The gateway then serves as a front-end connector to bridge all external procurement systems to an internal tier-2 common commerce engine as described in the next subsection.

2.3 Unified Commerce Server

With a unified commerce server strategy, Common Commerce Engine (CCE) is developed to consolidate the multiple commerce engines currently in use in IBM into a single hosting environment to execute the single IBM e-business strategy, to reduce operation and deployment costs, to reduce duplicate development efforts and to reduce time to release new features. CCE is composed of a set of reusable components that provide the core functions necessary to conduct the customer's e-commerce experience (i.e., shopping and purchasing). Currently, IBM is utilizing CCE to serve its business partners, large-enterprise e-sites, and IBM public site (ShopIBM[8]).

CCE is developed based on the IBM WebSphere Commerce products [9, 10], a comprehensive set of integrated software products for building, deploying, and growing e-businesses. Actually, CCE provides extensions above and beyond the base capabilities to allow a richer Web experience for the diversity of customers and business partners through personalized page presentations, tighter linkage with telesales and support, and robust system performance.

3. MASTER CATALOG AND ENTITLED PRICES

For a large enterprise supplier that has one-to-many relationship with buyers, a dedicated channel for each of its buyer may have to be established to promote and sell its products. In this case, pricing and terms and conditions are important parts of its online catalogs, and are the results after a thorough negotiation for the enterprise contract between the buyer and supplier organizations. For small number of customer enterprises, creating dedicated catalog for each enterprise is not an issue for the supplier. However, for large number of customer enterprises, scalability, administration and maintenance become problematic.

To overcome this limitation, CCE deploys only a single master catalog that includes products from all brands. To support a large set of enterprise customers in their dedicated store channel, the master catalog is then dynamically filtered to produce logical sub-catalogs with discounted prices. These sub-catalogs, referred as entitled catalogs, are produced based on selected customer groups with their entitlement and discount policies provided by their existing contracts with IBM as the supplier.

In fact, CCE provides a sophisticated mechanism to offer these entitled catalogs with various pricing and offerings that can be configured via complex configuration tools. For those offerings with simpler pricing models, the associated pricing terms are obtained along with the entitled catalog instance to localize the entitled price calculation within the commerce engine. With more complex pricing models, CCE allows the use of an external pricing service to obtain in real-time the appropriate entitled prices based on the associated contracts for the selected offerings.

Two general types of prices are defined in CCE: (1) *base price* and (2) *discounted price*. Base price is also called list price. Base prices for products are set up in master catalog and are common to all stores, while discounted prices are specific to buyer enterprises based on the their business contracts. Discounted price is further categorized into *fixed price discount*, *promotional discount* and *regular discount*. The pricing and discount rules for these types of prices are summarized below.

Fixed price discount is defined only for products, and is the price offered at a specific dollar value. Multiple fixed prices can be defined for products, and only the effective fixed price with the highest precedence at any given time will be in effect. If there is more than one effective fixed price with the same precedence, the lowest price will be in effect. Fixed prices, if defined, also take precedence over promotional or regular discount.

Promotional or regular discount is a single percentage discount defined for products or product groups (categories). Promotional discount has start and end dates, and is effective only between these dates, while regular discount has no expiration date. If a promotional or regular discount is defined at a category, the discount will be inherited by every child categories or products lower in this category sub-tree in the catalog unless there is a discount explicitly defined at any lower level. If both promotional and regular discounts exist for the same category or product, the promotional discount will take precedence over the regular discount.

In CCE, the final discounted prices for products are calculated based on the base prices and effective percentage discounts according to the dynamic algorithm as depicted in Figures 2 and 3. The detailed description of the algorithm is given in Appendix A.

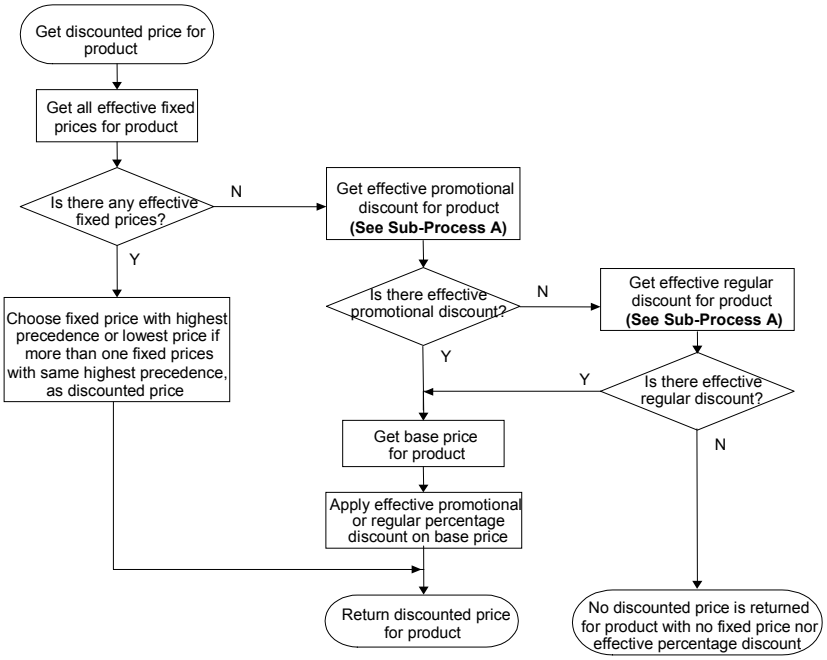

Figure 2: Pricing algorithm in Common Commerce Engine to get discounted price for a product

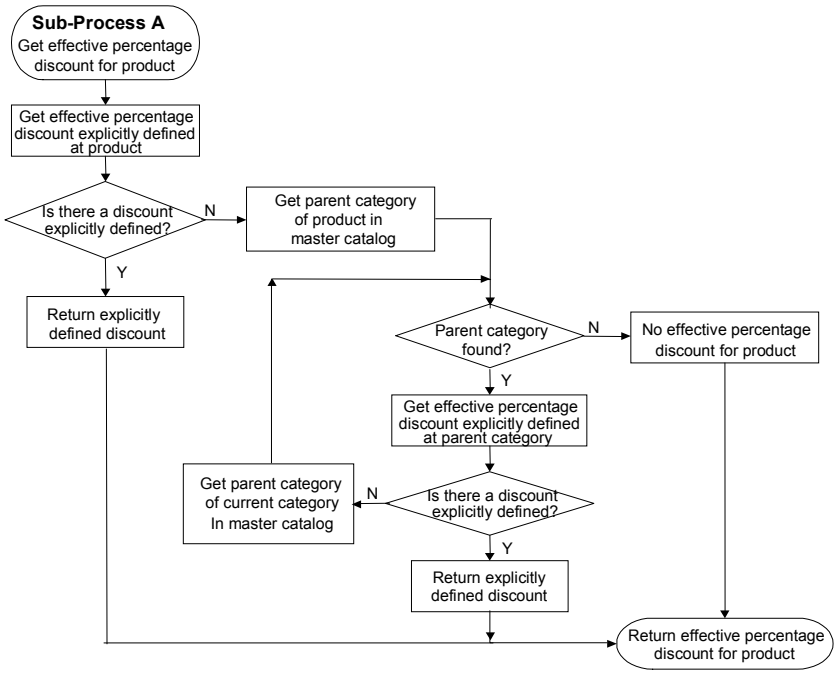

Figure 3: Sub-Process A for pricing algorithm in Common Commerce Engine to resolve promotional or regular percentage discount for a product

With dynamic entitled price calculations, when a buyer customer enters and navigates in a custom B2B store for shopping, they are essentially presented with a virtual catalog with entitled product prices. In this way, the commerce engine does not need to create and store a large number of physical catalogs for all the custom stores, thus reducing storage, administration and maintenance requirements to provide better flexibility and scalability.

4. PROCUREMENT SOLUTIONS

Software vendors such as such as Ariba, mySAP[11], Oracle[12] and CommerceOne[13], have introduced various electronic procurement systems to allow buyers and suppliers to link their systems together through Internet. There are two general types of e-procurement solutions of particular interests. The first type is called catalog punchout, which is usually implemented as a single-tier solution in which the supplier catalog in its Web site is enabled for direct access by customers' procurement systems for online shopping, and the resulting shopping cart is formatted and returned to the procurement system for approval. The other type is called local catalog, in which the supplier's catalog is extracted and formatted for import into an application that resides locally in the buyer's system.

For both of these solutions, there are a number of commercial protocols to be considered. In the next sub-sections, we discuss how to support both the remote catalog punchout and the local catalog solutions with a unified commerce server architecture.

4.1 Remote Catalog Punchout

Catalog punchout is an online procurement integration solution, that allows buyers to access supplier's Web site within the buyer's own procurement applications. The buyers leave ('punch out' from) their procurement system and go to the supplier's Web-based catalog to browse and to order products, while their procurement application transparently maintains connection with the Web site and gathers pertinent information. As such, buyers can search, select, and configure products for purchase in the supplier's Web site while bringing the shopping cart and purchase order information back into their own applications.

Punchout is a way for organizations to maintain control of their internal purchasing process without giving up the ability to access personalized catalog content and updated pricing and inventory information. It allows organization buyers to access more products and inventory from more suppliers in real-time, and to view updated product information that includes technical data and images that would be otherwise unavailable in an internal procurement system. Punchout also allows supplier organization to present their rich content already contained in their B2B storefront, and provides up-sell potential for market differentiation.

Examples of punchout vendors are Ariba, mySAP, and Oracle. Ariba punchout is based on cXML, which comprises a meta markup language and a protocol for data exchange between applications. mySAP and Oracle punchout solutions are based on Open Catalog Interface. There are also other similar punchout solutions based on Open Buying on the Internet[14].

An enterprise supplier such as IBM needs to support a large number of B2B stores for their enterprise customers. These enterprise customers may access the B2B stores through dedicated front-end portals or use their procurement systems to perform catalog punchout. To fulfill the requirement that the same entitled catalogs should be presented to the same set of B2B customers regardless of their choice of access method into the commerce site, we have implemented in our

solution a single sign-on with a universal session cookie (USC) and an automatic sign-on procedure in CCE to allow single sign-on from either front-end portals or from the tier-1 B2B gateway. The details of single sign-on mechanism with universal session cookie, and common return format in CCE are described in the next subsections.

4.1.1 Single Sign-On with Universal Session Cookie

Enterprise security within private B2B stores is an import issue to be considered here. To prevent unauthorized access to enterprise resources, secure portals or punchout networks have to implement some authentication mechanism through user identification and password logon process. In a multiple tier system, there may be a logon process for each system; hence users may have to perform logon process more than once. This may have caused much inconvenience for users to use multiple user identifications and passwords to access systems in order to perform their commerce activities.

To support direct shopping and punchout requests from both secure portals and B2B Gateway, we have implemented in CCE a single sign-on mechanism based on a trust entity received from the trusted front-end system. This single sign-on mechanism consists of an automatic sign-in procedure and an encrypted universal session cookie (USC) valid within the enterprise domain to be passed exchanged within the internal systems.

The USC is first created in the requesting host system, i.e., the commerce front-end portal or the B2B Gateway connector, upon successfully authenticating the incoming user. This USC is then forwarded to CCE for requesting access to B2B store or catalog punchout support in the commerce engine. The required information for the user, user organization, and list of B2B stores authorized for this user organization is embedded in the USC cookie value as name-value pairs separated by well-defined delimiters. This required information is retrieved from an organization and user profile database stored in the requesting host system or in a remote LDAP server.

Besides B2B Gateway and secure portals, CCE, based on IBM WebSphere Commerce products, has also a build-in and sophisticated authentication and authorization mechanism. Resources in CCE are protected based on fine-grained access-control policies related to user role and actions. To access the proper resources, organization users need to be authenticated and authorized through a secure logon process. In our 2-tier solution, since users are already authenticated and authorized by a trusted first-tier B2B Gateway system or portals within the enterprise, this logon process can be made transparent by using a common, internal sign-in procedure.

The automatic sign-in procedure first checks whether the user in the request has an entry in its membership records. If no registration record is found, CCE will perform a new registration, and establishes a new session for the user. The required registration parameters are retrieved from the USC value. If an existing registration record is found, CCE re-establishes the existing session for the user, and retrieves the contents of existing shopping cart for edit or check out process.

An additional punchout indicator is also added in the USC to differentiate normal portal access session from a punchout session. For case of normal shopping session, the portal shopper is asked to provide payment methods and shipping information during check out. In contrast, for the later case of punchout session, the procurement shopper is allowed to submit the shopping cart to be returned to the buyer's procurement system as a quote for approval. Another usage of

this punchout indicator is for the personalization of look-and-feel and selection of display pages for the punchout shoppers during shopping session in the B2B stores.

4.1.2 Common Format for Return Shopping Cart

In our 2-tier punchout solution, the entire submission process is a two-step process. CCE first submits the contents of the shopping cart to B2B gateway, which then performs a protocol-dependent format conversion before submitting the resulting quote back to the requesting procurement system. To simplify format conversion, CCE has implemented a common HTML form similar to the HTML version in Open Catalog Interface (OCI) to hold the quote information. All user and organization information, and line items of the shopping cart are placed in hidden fields as name-value pairs in the form. The quote information consists of supplier identification, buyer identification, currency, total price, order item SKU, item price, item quantity, item description, manufacturer and contract information.

The use of a simple HTML form similar to OCI HTML version for return quote in CCE is based on its simplicity in implementation. One may chose to implement a XML version of message to place the contents of the shopping cart here. However, there still need a redundant step to parse and convert this proprietary XML message into a protocol-dependent format at the B2B gateway for the requesting procurement system.

4.2 Local Catalogs

In an e-procurement solution with local catalogs, physical catalogs with entitled prices are considered to be one of the most critical components. Local catalogs provide buyer organizations with a mechanism to make controlled, yet non-restrictive purchases from term-contracts representing negotiated pricing with approved vendors. Typically, the entitled catalog is first extracted from a supplier site and is then imported into a local procurement system of the catalog-receiving enterprise for use.

4.2.1 Practical Constraints

Enterprise supplier such as IBM needs to provide entitled catalogs for customers requesting local catalog on a daily basis. CCE needs to implement an extraction process to satisfy these customers. However, in CCE architecture, there is no physical entitled catalog stored in database for each custom B2B store, only a single master catalog. It utilizes a real-time, dynamic pricing algorithm to resolve the entitled prices for products to be presented to users of custom B2B stores. Particularly, the pricing algorithm requires multiple accesses to database to retrieve and resolve the percentage discounts for product, product category and parent category corresponding to a given enterprise contract (see pricing algorithm given in Appendix A). In addition, multiple contracts for a given enterprise customer can exist, and the best prices must be determined.

Practically, CCE has a daily process to update the content of its own master catalog and list prices. This update process has been optimized and takes approximately a certain fixed amount of time. Considering the limited daily time window to accomplish the entire update and publish process, a key challenge facing CCE is its ability to allow extraction of a large number of entitled catalogs with least amount of time after the content of master catalog is updated. We need to develop and deploy an efficient process to resolve and extract the entitled prices for products in batch operations.

4.2.2 Prior Extraction Procedure

A straight-forward method for the creation of entitled catalogs is to apply the CCE internal pricing algorithm as given in Appendix A for the calculation of discounted prices. This method is simple and requires least amount of time to develop. Basically, the method starts from a full list of products in the master catalog and utilizes the CCE pricing algorithm to resolve their effective percentage discount with a given contract corresponding to an enterprise buyer organization. The resulting percentage discount is then applied to the base price of the product to return a discounted price. However, it is found that this method requires a large amount of computation time in order to create the set of entitled catalogs for IBM large-enterprise customers, and is not quite optimal for deployment.

Basically, the CCE pricing algorithm is developed to support dynamic, real-time calculation of discounted prices for products listed on a given Web page of the commerce sites. They follow a bottom-up catalog-tree traversal approach to resolve the effective percentage discount at each upper sub-category level for a given product at a time. This approach is seen to perform repetitive and unnecessary traversals of the same upper categories for a set of products belonging to the same category. Since each traversal of a tree node translates to a number of executable commands and database accesses, the performance degrades rapidly for cases where the percentage discounts are only defined at higher levels of categories in the catalog tree.

For example, consider a worst case of an entitled catalog with total of P products, and a uniform N level deep of product categories, and assume that a single percentage discount is defined only at the top category level, and that no other discount is defined at other lower category or product levels. The bottom-up tree traversal approach introduces a sub-total of P traversals through all categories at the same level within the catalog tree, and hence a total of (N+1)P traversals for the entire catalog.

To show that the multiple (N times in the above example), repetitive traversals of the upper level categories can be minimized, we have proposed to use a top-down tree traversal approach to first resolve the potential effective percentage discount at lowest categories. The details of this approach are given in next sub-section.

4.2.3 Proposed Extraction Procedure

To optimize the number of traversals for each upper category, we develop an efficient algorithm following a top-down tree traversal approach to first resolve the potential effective discount percentage for each category at the lowest category level. This is done by traversing the catalog tree from top category down to lower sub-categories using a breadth-first traversal.

Basically, we want to first build the Category/Effective-discount maps containing (Category, Effective discount) pairs for all categories from top and down to the lowest category levels. We achieve this by propagating and applying all upper level effective discount values to lower levels according to the following rule: If the category in the catalog tree has a percentage discount, this discount will be compared with that of the parent category. The best of the two becomes the effective percentage for the category and is stored to the category/effective-discount map as (Category, Effective discount) pairs.

We need to apply the above rule separately to both the promotional percentage discounts and the regular percentage discounts to obtain a category/effective-promotional-discount map for the promotional discount, and a category/effective-regular-discount map for regular discount. The

sub-set of these maps containing (category, effective discount) pairs of the immediate parent categories of the products is used as the base for resolving the discounted prices for all products. The procedures are depicted in Figures 4 and 5, and are described below:

(1) For each product entry, if it has some effective fixed prices, the fixed price with the highest precedence will be the discounted price for the product entry. If there are multiple fixed prices with the same highest precedence, the lowest fixed price will be the discounted price.

(2) If there is no effective fixed price for the product entry, look for the effective percentage discount for the product entry using Process B as described below:
    (a) First, look for the promotional or regular discount explicitly defined for the product. If a discount is found, it will be the effective percentage discount for the product. Note that promotional discount takes precedence over regular discount if both found.
    (b) If no regular discount found at the product entry level, resolve the effective percentage discount for its immediate parent category by first checking the Category/Promotional-discount map. If a promotional discount is found, it will be the effective percentage for the product.
    (c) If no promotional discount is found from Category/Promotional-discount map for the immediate parent category, check for regular discount for its immediate parent category from Category/Regular-discount map. If a regular discount is found, it will be the effective percentage for the product.

(3) After getting an effective percentage discount, it will be applied to the base price of the product to return the discounted price. If there is neither effective fixed price nor effective percentage for the product entry, it means the product is not entitled and no discounted price is returned.

Consider the same example for the worst case in the creation of an entitled catalog given in Section 4.2. For a catalog with P number of products, a uniform N level deep of product categories, and a total of M sub-categories within the catalog tree, the number of traversals to upper categories in the catalog tree using our proposed top-down tree traversal approach is reduced to M traversals only. Hence, the total number of traversals requires to create the entitled catalog is (P+M) traversals. Compared to the result of Section 4.2 example, our proposed approach can potentially achieve a factor of (N+1)P/(P+M) or approximately (N+1) when the number of products is large.

For typical large catalogs with discount values defined randomly at different levels of categories, it can be shown that the top-down tree traversal approach still outperforms the bottom-up tree traversal approach in most of the cases. For the best case where all discounts are explicitly pre-defined at lowest categories, our proposed approach requires only an extra M number of calculations. Typically, this number corresponds to a relatively small number in comparison with the total number of products, hence only a minor increase in calculation time. In fact, by applying our proposed approach to the creation of a set of entitled catalogs for IBM large-enterprise customers, a substantial improvement in total calculation time has been achieved.

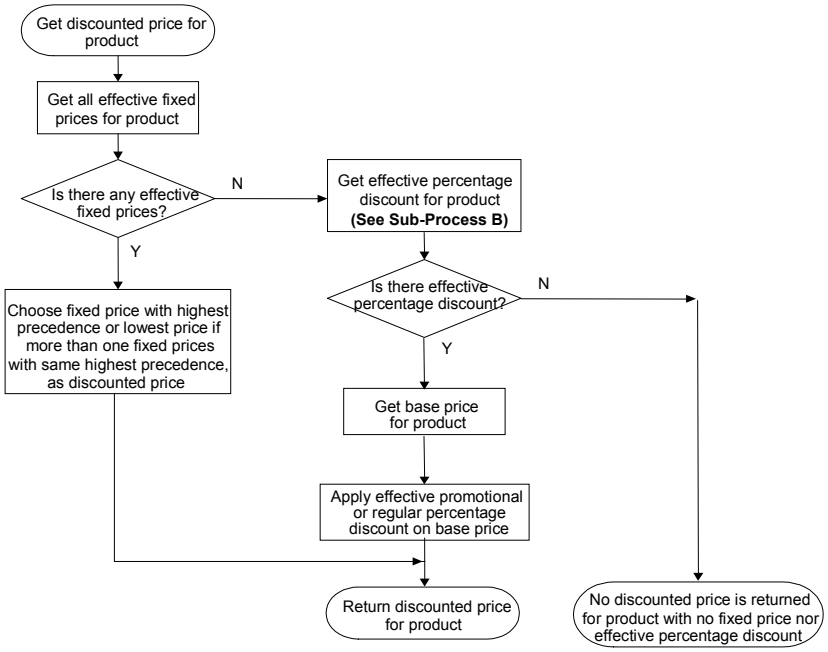

Figure 4: Proposed pricing algorithm for catalog extraction procedure to get discounted price of a product

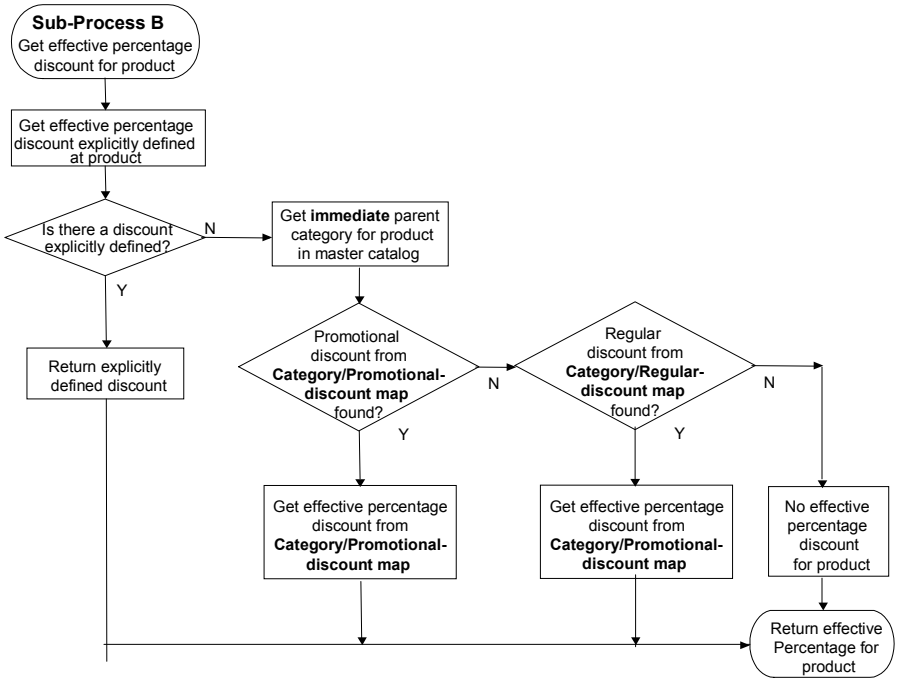

Figure 5: Sub-Process B of proposed pricing algorithm for catalog extraction procedure to get discounted price of a product

5. SUMMARY AND CONCLUSION

In this paper, we describe our practical implementation experience for IBM, on how to use a unified commerce server architecture to support and accommodate various shopping paradigms including B2B direct catalog shopping, B2B procurement with remote catalog punchout and B2B procurement with local catalog. We present a 2-tier commerce architecture with a tier-1 B2B gateway system to handle external protocol-dependent interfaces, and a tier-2 common commerce engine to provide core commerce functions for shopping and order creation. We explain how a single master catalog can be dynamically filtered to provide personalized, entitled catalogs for custom B2B stores. We describe how the Common Commerce Engine can handle various punchout protocols with a common implementation of unified return format and a single sign-on based on a universal session cookie. We describe the issues and efficiency considerations for real-time entitled pricing lookup for Web-based shopping versus batch pricing calculations for mass extraction of entitled catalogs. We also present an efficient procedure and process for batch extraction operations based on a pre-computation of discounts at category levels using a breadth-first catalog tree traversal method.

APPENDIX A: Detailed Description of CCE Dynamic Pricing Algorithm

(1) If a product entry has some effective fixed prices, the fixed price with the highest precedence will be the discounted price for the product entry. If there are multiple fixed prices with the same highest precedence, the lowest fixed price will be the discounted price.

(2) If there is no effective fixed price for the product entry, look for the effective percentage discount for the product entry. First, look for the effective promotional discount, and apply the following recursive process:
   a) If there is an effective promotional discount at the product entry level, this discount will be the effective percentage for the product entry
   b) If no effective promotional discount found at the product entry level, look for one at category level. If found, it will be the effective percentage and proceed to step(4).
   c) If no effective promotional discount found at category level, look for one up in the catalog tree. If found, it will be the effective percentage and proceed to step(4).
   d) If no effective promotional discount found at parent group level, look for one further up in tree by repeating and following step (2c) above.
   e) If no effective promotional discount found at top level, go to step (3).

(3) If no effective promotional discount found, look for the effective regular discount by applying the following recursive process:
   a) If there is a regular discount at the product level, this discount will be the effective percentage for the product.
   b) If no regular discount found at the product level, look for one at the category level. If found, it will be the effective percentage and proceed to step (4).
   c) If no regular discount found at the category level, look for one at its parent level. If found, it will be the effective percentage and proceed to (4).
   d) If no regular discount found at parent level, look for one further up in the catalog tree by repeating and following step (3c) above.
   e) If no regular discount found at top level, there is no effective percentage for the product.

(4) Calculate the discounted price via applying the effective percentage on the base price with the highest precedence for the product.

(5) If there is neither effective fixed price nor effective percentage for the product, it means the product is not entitled. So no discounted price is returned.

REFERENCES

[1] Nahid Jilovec, "The A to Z of EDI and Its Role in E-Commerce", second edition, Duke Press, 1998.

[2] T.C. Chieu, F. Pinel, S. Fu & J-S. Yih, "Unified Solution for Procurement Integration and B2B Stores", The 5th International Conference on Electronic Commerce, ICEC 2003, p.61, October 2003, Pittsburg, PA.

[3] "Commerce Extensible Markup Language (cXML) User Guide", Version 1.2, Ariba Inc., 2001. http://xml.cxml.org/current/cXMLUsersGuide.pdf.

[4] RosettaNet standards, RosettaNet Global Industry Consortium. http://www.rosettanet.org/RosettaNet/Rooms/DisplayPages/LayoutInitial?container=com.webridge.entity.Entity[OID[5F6606C8AD2BD411841F00C04F689339]].

[5] OAG business object documents, Open Applications Group Inc. http://www.openapplications.org/.

[6] "Open Catalog Interface (OCI)", Release 2.0B, SAP AG, 2000. http://www.sap.com/partners/icc/scenarios/pdf/b2b_oci_20b.pdf

[7] S.S. Fu, S-K. Chen, J-S. Yih, F. Pinel and T.C. Chieu, "Web-based Sell-side Commerce Aggregation", International Conference on Internet Computing, pp. 303-310, June 2002.

[8] IBM public site (ShopIBM), IBM Corp. http://www.ibm.com/products/us/.

[9] IBM WebSphere Commerce Business Edition, IBM Corp. http://www-3.ibm.com/software/webservers/commerce/wc_be/

[10] D. M. Dias, S. L. Palmer, J. T. Rayfield, H. H. Shaikh, T. K. Sreeram, "e-Commerce Interoperability with IBM's WebSphere Commerce Products", IBM Systems Journal, Vol. 41, No. 2, pp. 272-286, 2002.

[11] SAP procurement, SAP AG. http://www.sap.com/solutions/bi/procurementinsightpackage/

[12] Oracle procurement, Oracle Corp. http://www.oracle.com/applications/B2B/internet_procurement/index.html?intro.html

[13] Commerce One solutions, Commerce One Operations Inc. http://www.commerceone.com/solutions/srm_tech.html.

[14] "Open Buying on the Internet" Technical Specifications, Release 2.0, The Open Buying on the Internet (OBI) Consortium, 1999. http://www.commerce.net/oldwebsite/research/technology-applications/1999/99_28_r.html