e-Transformation Technologies: Case Studies and The Road Ahead – A Value Chain Perspective

William Kwok-Wai Cheung
Department of Computer Science
Hong Kong Baptist University
Kowloon Tong, Hong Kong

Abstract
e-Transformation technologies, for the past few years, have been evolving towards the goal of information integration and system interoperability. While there is no doubt that interoperable systems can enable many value-added functions and provide new business opportunities, whether the current development trend can really lead to some real impacts on people's daily lives is still a commonly asked question. In this paper, we first describe a value chain of e-transformation that starts from the availability of data from heterogeneous sources and stops at the ultimate goal of autonomous and context-aware computational services provision. Based on this value chain, we first describe in details two of our recently developed projects, namely e-Bus Planner and e-Food Marketplace for demonstrating the promises that can be provided by information integration applications. Then, we address the limitations of the projects from the end-user perspective and argue that the value adding process should further be advanced along the chain with the use of Web intelligence techniques for analyzing Web-based social networks. The ultimate objective is to benefit the end-users more directly and immediately.

Keywords: e-Transformation, value chain, information extraction, information integration, Web intelligence

1. Introduction

e-Transformation is here defined as the transformation of some operational processes caused by the incorporation of advanced information technology, where paradigm shift and process reengineering are usually resulted. One of the best examples is the evolution of e-commerce from e-cataloging, to e-tailing, to e-business and eventually to e-enterprise, causing a lot of companies to reconsider their e-commerce strategies almost every year. e-Transformation technologies, or simply called e-technologies, are referred to as those technologies which facilitate the transformation to happen efficiently and effectively. In many cases, the e-technologies are common for different problem domains (say e-commerce, e-learning, e-government). Some examples include on-line payment, search engines, middleware for system interoperability, use of ontology, etc.

After more than a half-decade of e-transformation, there are many important lessons learnt from the market. For example, the first-mover strategy does not always work. Also, running some processes on-line does not necessarily imply that the involved cost is low [1]. Eventually, gaining competitive advantage in the Internet is found to be not easy and identifying the true values that different e-technologies can provide to users becomes especially important.

The e-transformation value chain in fact can roughly be divided into a number of important value-adding phases: (1) information dissemination, (2) information extraction and

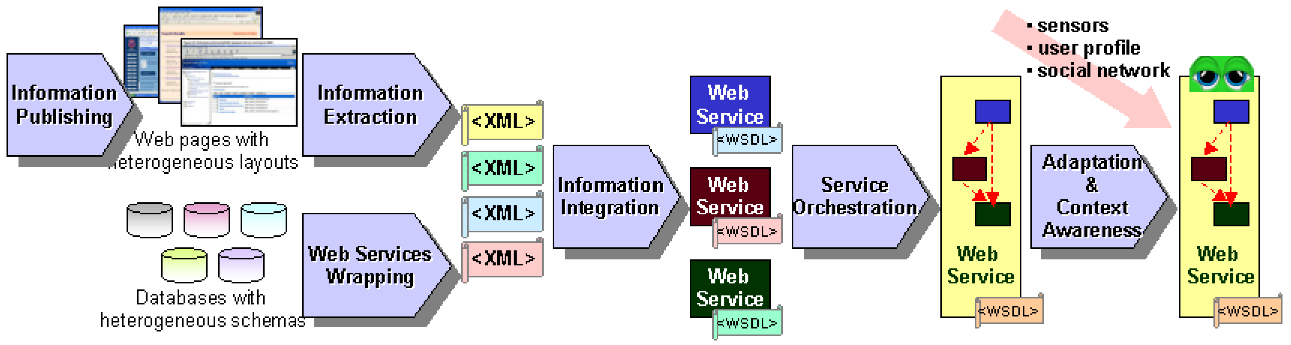


Figure 1 The e-Transformation Value Chain.

integration, (3) services orchestration, and eventually to (4) problem solving (in a distributed and wireless environment). A more detailed overview and analysis are provided in Section 2. According to the experience of many not so successful e-transformation related examples (no matter in e-learning, e-government, e-business, etc.), one crucial factor lies in the fact that the end-users have not been effectively satisfied by the existing e-services. Recently, there have been a lot of effort in developing techniques and applications which are adaptive to user behaviors (autonomous/ubiquitous computing) and environmental context (context-aware computing), with the hope of adding high enough value so that the end-users will eventually decide to get over the barrier and switch from their traditional modes of operation to the ones enabled by the new e-technologies. In addition, we believe that in the last mile of value adding process, the value added functions should be targeted at not only providing new opportunities but also making them to be able to benefit the end users directly.

In this paper, we first provide an overview of the e-transformation value chain in Section 2. Then, we describe in detail two of our recently implemented prototypes for illustrating part of the e-transformation mediation path in Section 3 and 4. Then, in Section 5, we argue that in order for those related systems to have high impact to end-users, *Web Intelligence* (WI) techniques should be integrated into the devices commonly used by ordinary users, e.g., personal information management (PIM) systems, so that the underlying *social intelligence* can readily be collected to achieve the high-level goals set by the end-users in an autonomous and adaptive manner. Section 6 concludes the paper.

2. The e-Transformation Value Chain

With the advent of hardware technologies for human computer interaction (e.g., the use of touch-screens, pen-based computers, speech recognizers) as well as the ever-decreasing production cost, the digital divide due to the proficiency in operating computing devices is gradually closing up. However, penetration of information technology into our daily lives still requires "smart" enough software systems which can take some easily specified high-level problems as input and solve them based on up-to-minute information without further human intervention (unless for necessary clarifications). For example, we may be looking forward to the day when we can ask our PDA (by pressing a button) or an information kiosk (by presenting your smart card) for a "How to take a bus home?" service and can get directly the bus route numbers available in a nearby bus stop with the traffic conditions taken into consideration. For another business-related example, one can ask his/her desktop computer for a "Want to place orders to suppliers for stock refilling" service (saying by clicking a button on the desktop) and the computer returns you a suggested purchasing order after negotiating with the suppliers' software agents. Of course, we know that we are still far from the day with the aforementioned examples happened in our daily lives. In this section, we try to look at the way to achieve the goal using an e-transformation value chain.

The e-transformation value chain starts from Web pages of heterogeneous sources to value-added services like personal assistants and involves various e-technology architectures and tools to make the value keep increasing along the chain. Figure ? provides a pictorial view of the value chain. For each phase of the chain, discussion related to the information/functional characteristics, the tools required, and the value being added will be provided in the following.

2.1 Heterogeneous Information Sources

2.1.1 Sources as HTML Pages

Characteristics: HTML pages contain *semi-structured* as well as *unstructured* information. They can be static or dynamically generated (e.g., database driven). To contrast, static Web pages is normally less structured (or sometimes almost unstructured) while dynamic pages, as generated by programs, should normally be more structured.

Tools: To support dynamically generated Web pages, database connectivity via Web programs (e.g., DBI, JDBC, etc.) is needed.

Value added: On-line users can get up-to-minute information by visiting particular pages and locating the information.

2.1.2 Sources as Web Services

Characteristics: Web services is an emerging standard of on-line APIs. They can be simply information provision services (in XML format through SOAP messaging) or together with some computational processing involved. The *granularity* and the *structuredness* of the accessible information or services are determined by the Web services' "methods" whose details are normally publicized in a repository (e.g., UDDI) using Web Services Description Language (WSDL).

Tools: Matchmaking engines assisting the information providers to match their own database schema to some standard one are needed to facilitate effective cross-institution (e.g., B2B) information exchange.

Value added: Information (structured data) can now be made available to other computer programs, which in turn enables systems interoperability and integration within or between enterprises. In addition, by introducing a presentation layer, the structured information in XML format can easily be "transcoded" into other mark-up languages like XHTML, WML, etc. for human-friendly presentation.

2.2 Web Page Wrapper - Information Extraction

2.2.1 Wrapping HTML Pages

Characteristics: Information extracted from Web pages using wrappers can then be made available to other computer programs via, say, Web services.

Tools: Owing to the different structuredness of the data embedded on Web pages, wrappers with different capacities are needed. To extract tabulated data, wrappers capable of analyzing the grammatical structures of HTML tags are needed [2]. For some textual information items, e.g., customers' comments, more sophisticated techniques capable of analyzing the natural language structures are needed [3]. Note that this extraction process should normally be done by some external brokers, rather than the information providers, for providing further value-added functions (see Section 2.3).

Value added: With the use of wrappers, we can enable a Web page information source into a Web Service enabled one, and thus the users can gain all the values as described in Section 2.1.2.

2.3 Composite Service I – Information Integration

Characteristics: Information of different structures (e.g., due to the use of different XML tags or XML schemas) are semantically integrated.

Tools: Tools that can perform matching of related XML schemas with possibly missing items are needed (as it is common that heterogeneous information sources contain information details to different extents). Domain-specific ontologies are commonly required. Related mark-up language standards (OWL), editing tools and inference engines are also needed.

Value added: Aligning semantically corresponding information from heterogeneous sources can save on-line users the effort in switching between different Web sites and comparing in detail the embedded information, which is commonly needed for comparison shopping.

2.4 Composite Service II – Service Orchestration

Characteristics: Semantic information can be interchanged and semantic services can be interacted to solve a problem via orchestrating the available Web services.

Tools: Mechanisms to publish and discover services using their semantics, including their functionalities, current capacities, etc. are needed. Workflow engines for coordinating multiple services for executing a task are also important. QoS composition can guarantee the promised service quality. [4,5]

Value added: Tasks requiring coordination of multiple types of services can thus be delegated. Most of the high-level tasks initiated by users can effectively be solved directly using composite services of this type, without the need of taking care the time-consuming coordination details.

2.5 Adaptive and Context-Aware Applications

Characteristics: The solutions provided by composite services can be adaptive to the current context and the user profiles.

Tools: We need to acquire and represent the current context (location, traffic conditions, time zones, etc.), and thus sensors are sometimes needed. The quality of the context-aware applications depends heavily on the sensors' accuracy. Thus, different sensor fusion or denoising techniques are needed. Also, we need to acquire and represent the user profiles (e.g., interest, credit card information, home address). Personalization and social intelligence techniques are then needed for directing the adaptation of the underlying services. [6]

Value added: Applications become autonomous and proactive which can free users from initiating processes and tuning them to suit the current context and their current interest.

In the next two sections, we describe in detail two of our recently developed prototypes that are related to the two examples we mentioned at the beginning in Section 1. The first project is called *e-Bus Planner* which takes information posted on HTML pages (semi-structured heterogeneous data) as input, and then wraps and transcodes the information into XML format (information extraction), and eventually integrate the information to support bus trip planning (information integration). The second project is called *e-Food Marketplace* which takes data records in database (structured data) as input, and then assist the user to derive Web Services for information provision (schema matching), and eventually support virtual catalogs in e-Marketplaces (information integration).

3. e-Bus Planner

Public Bus Network in Hong Kong is well developed, which connects most urban and rural areas. The network is massive and complicated. Planning a trip on this public bus network can be a non-trivial and time-consuming task, especially for tourists. While bus routes related information has been made available on the Web by different bus services operators, those Web sites are not interoperable, making bus trip planning involving multiple operators almost impossible. In this project, we have built a Web-based information assistant called *e-Bus Planner* to support bus trip planning. e-Bus Planner extracts and integrates online bus routes information of different bus services operators to support bus trip planning. The information assistant has a set of tools for managing the underlying processes for the just-in-time trip planning, including data retrieval, data conversion as well as integrated path searching modules where bus trips involving multiple bus routes operated by different operators can easily be identified.

3.1 System Overview

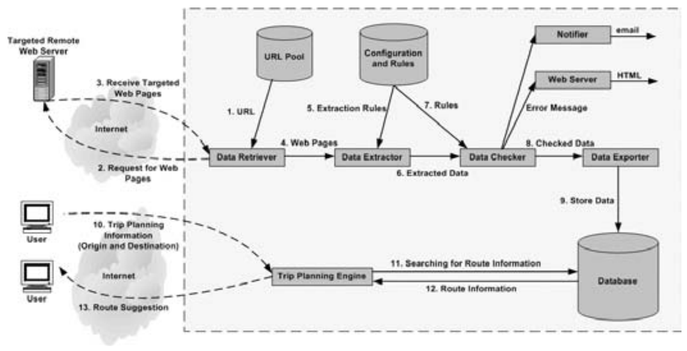


Figure 2 The system architecture of e-Bus Planner.

Figure 2 gives the overall system architecture of e-Bus Planner. The system is built to support two main processes, namely *information extraction* and *information integration*. The former process is enabled by four major components, namely *Data Retriever, Data Extractor*, *Data Checker* and *Data Exporter*. It is supposed to be run by the system administrator of e-Bus Planner. The latter process is enabled by a major component – *Trip Planning Engine*. A user starts planning a trip by providing the origin and the destination to the Path Searching Engine for suggestions of bus trips with minimum transportation cost. Other relevant information such as attraction points, restaurants, and fast food shops near the destination can also be displayed to the user.

3.2 Information Extraction from Multiple On-line Sources

Web pages are human readable with attractive layouts. Useful information are typically embedded in the some tables in the pages possibly surrounded by graphics. The goal of information extraction is to collect specific information (e.g., public bus information in our case) from particular pages of some Web sites (e.g., routes information pages maintained by multiple bus services operators). The extraction part of eBus Planner is designed according to the Myllymaki's framework [7]. In the following, we describe the components involved in the context of bus trip planning.[1]

3.1 Data Retriever

Data Retriever is designed to communicate with target remote Web servers. It retrieves Web pages from Web servers with their URLs pre-defined by the administrator. The URLs of most Web pages published by operators are in regular patterns that can be easily identified. Some examples are listed in Figure 3.

```
Route Number 1, Kowloon Motor Bus
http://www.kmb.com.hk/english.php?
page=search&prog=route_no.php&route_no=1
Route Number 2, New World First Bus
http://www.nwfb.com.hk/eng/routesearch/
routesearch06s.asp?v_first=2
```

Figure 3 The URL patterns for retrieving routes information from two different bus services operators.

3.2 Data Extractor

Data Extractor is designed to extract data from specific locations of the retrieved Web pages. It consists of two

[1] The design of the proposed system in fact can be easily ported to other applications.

major functions - Data Preparation and Data Extraction.

3.2.1 Data Preparation

Web pages are written in HTML. However, it has been well known that most of the Web documents found on the World Wide Web are not conformed to the official HTML specifications recommended by the World Wide Web Consortium (W3C). Reading and processing ill-formed HTML documents may lead to a high error rate in data extraction.

An ill-formed HTML document fragment

```
<p>here is an emphasized <em>paragraph.</p></em>
```

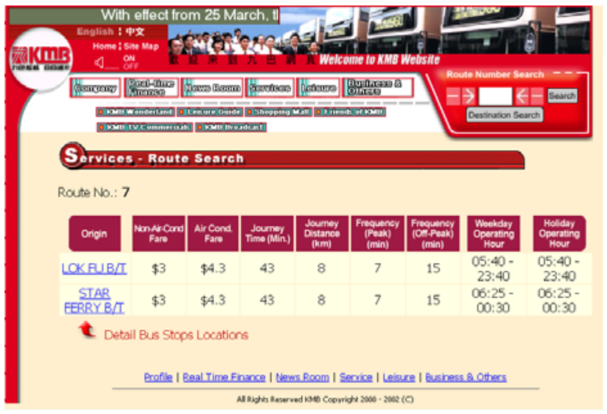
After converting it to a well-formed document fragment

```
<p>here is an emphasized <em>paragraph</em>.</p>
```

Figure 4 Examples of ill-formed HTML documents.

Data preparation is to repair ill-formed HTML documents and translate them to well-formed ones. The package called JTidy [8] is used for the convertion as shown in Figure 4 and the output is in the format of XHTML which is in XML syntax. This implies that all XML tools such as Extensible Stylesheet Language (XSL), XSL Transformation (XSLT) and XPath can be used for the subsequent extraction processes.

3.2.2 Data Extraction

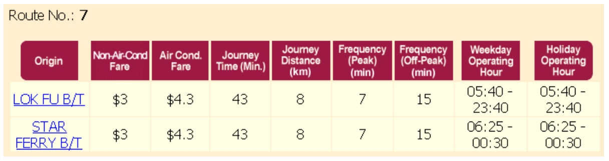


Route No.: 7

Origin	Non-Air-Cond Fare	Air Cond. Fare	Journey Time (Min.)	Journey Distance (km)	Frequency (Peak) (min)	Frequency (Off-Peak) (min)	Weekday Operating Hour	Holiday Operating Hour
LOK FU B/T	$3	$4.3	43	8	7	15	05:40 - 23:40	05:40 - 23:40
STAR FERRY B/T	$3	$4.3	43	8	7	15	06:25 - 00:30	06:25 - 00:30

Detail Bus Stops Locations
Profile | Real Time Finance | News Room | Service | Leisure | Business & Others
All Rights Reserved KMB Copyright 2000 - 2002 (C)

Figure 5 Target Web Page (Source: Kowloon Motor Bus).

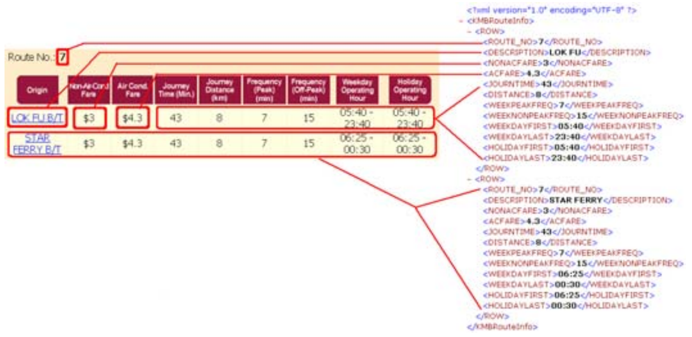


Route No.: 7

Origin	Non-Air-Cond Fare	Air Cond. Fare	Journey Time (Min.)	Journey Distance (km)	Frequency (Peak) (min)	Frequency (Off-Peak) (min)	Weekday Operating Hour	Holiday Operating Hour
LOK FU B/T	$3	$4.3	43	8	7	15	05:40 - 23:40	05:40 - 23:40
STAR FERRY B/T	$3	$4.3	43	8	7	15	06:25 - 00:30	06:25 - 00:30

Figure 6 Useful Information.

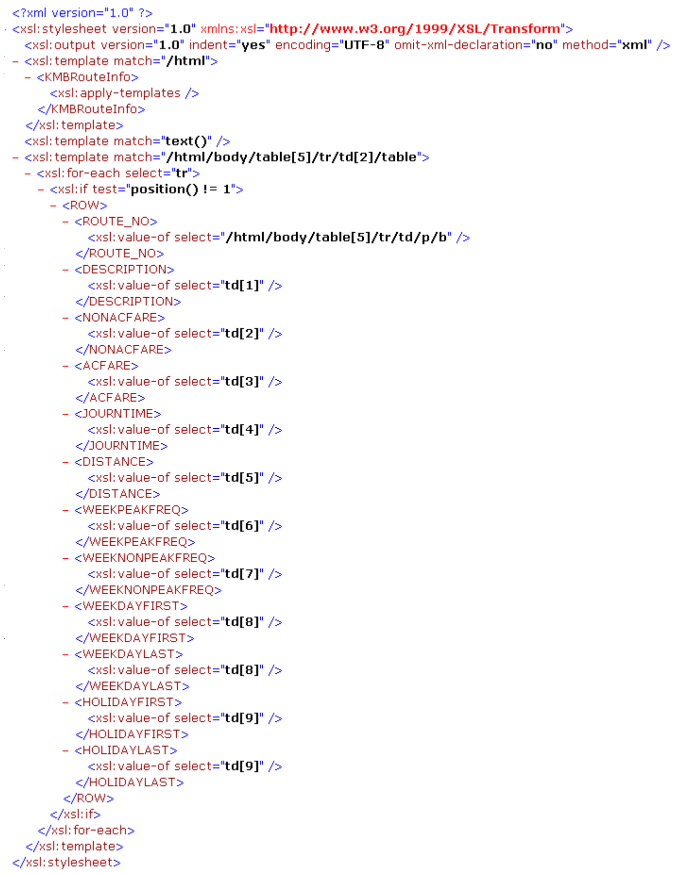
Figure 5 shows the Web page published by one of the operators. Multimedia elements such as graphics, animations and audio clips form part of the page to enrich the content for human readers. Bus route information is embedded in the Web page and the corresponding portion is shown in Figure 6. They are to be extracted out and stored in the database to support subsequent trip planning.



```
<?xml version="1.0" encoding="UTF-8" ?>
- <KMBRouteInfo>
  - <ROW>
      <ROUTE_NO>7</ROUTE_NO>
      <DESCRIPTION>LOK FU</DESCRIPTION>
      <NONACFARE>3</NONACFARE>
      <ACFARE>4.3</ACFARE>
      <JOURNTIME>43</JOURNTIME>
      <DISTANCE>8</DISTANCE>
      <WEEKPEAKFREQ>7</WEEKPEAKFREQ>
      <WEEKNONPEAKFREQ>15</WEEKNONPEAKFREQ>
      <WEEKDAYFIRST>05:40</WEEKDAYFIRST>
      <WEEKDAYLAST>23:40</WEEKDAYLAST>
      <HOLIDAYFIRST>05:40</HOLIDAYFIRST>
      <HOLIDAYLAST>23:40</HOLIDAYLAST>
    </ROW>
  - <ROW>
      <ROUTE_NO>7</ROUTE_NO>
      <DESCRIPTION>STAR FERRY</DESCRIPTION>
      <NONACFARE>3</NONACFARE>
      <ACFARE>4.3</ACFARE>
      <JOURNTIME>43</JOURNTIME>
      <DISTANCE>8</DISTANCE>
      <WEEKPEAKFREQ>7</WEEKPEAKFREQ>
      <WEEKNONPEAKFREQ>15</WEEKNONPEAKFREQ>
      <WEEKDAYFIRST>06:25</WEEKDAYFIRST>
      <WEEKDAYLAST>00:30</WEEKDAYLAST>
      <HOLIDAYFIRST>06:25</HOLIDAYFIRST>
      <HOLIDAYLAST>00:30</HOLIDAYLAST>
    </ROW>
  </KMBRouteInfo>
```

Figure 7 Useful routes info. extracted from a Web page.

Figure 7 shows the input and output of the Data Extraction process. After the HTML page is converted to XHTML format, it can readily be represented as a Document Object Model (DOM) tree. XML related technologies, i.e., XPath, XSL and XSLT as mentioned above, can then be used for extracting data from the tree. In particular, data extraction rules written in XSL format are fed into XSLT to convert the retrieved XHTML document to an XML document of pre-set format. The design of the XSL documents is customized according to the layout and structure of target Web pages (see Figure 8 for an example). Different Web pages structures require different XSL documents in order to correctly extract data from their heterogeneous layouts. Once the layout design of a Web page is changed, the extraction process will easily fail. So, it is important to create an XSL document robust enough to tolerate a small amount layout change.



```
  <?xml version="1.0" ?>
- <xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
    <xsl:output version="1.0" indent="yes" encoding="UTF-8" omit-xml-declaration="no" method="xml" />
  - <xsl:template match="/html">
    - <KMBRouteInfo>
        <xsl:apply-templates />
      </KMBRouteInfo>
    </xsl:template>
    <xsl:template match="text()" />
  - <xsl:template match="/html/body/table[5]/tr/td[2]/table">
    - <xsl:for-each select="tr">
      - <xsl:if test="position() != 1">
        - <ROW>
          - <ROUTE_NO>
              <xsl:value-of select="/html/body/table[5]/tr/td/p/b" />
            </ROUTE_NO>
          - <DESCRIPTION>
              <xsl:value-of select="td[1]" />
            </DESCRIPTION>
          - <NONACFARE>
              <xsl:value-of select="td[2]" />
            </NONACFARE>
          - <ACFARE>
              <xsl:value-of select="td[3]" />
            </ACFARE>
          - <JOURNTIME>
              <xsl:value-of select="td[4]" />
            </JOURNTIME>
          - <DISTANCE>
              <xsl:value-of select="td[5]" />
            </DISTANCE>
          - <WEEKPEAKFREQ>
              <xsl:value-of select="td[6]" />
            </WEEKPEAKFREQ>
          - <WEEKNONPEAKFREQ>
              <xsl:value-of select="td[7]" />
            </WEEKNONPEAKFREQ>
          - <WEEKDAYFIRST>
              <xsl:value-of select="td[8]" />
            </WEEKDAYFIRST>
          - <WEEKDAYLAST>
              <xsl:value-of select="td[8]" />
            </WEEKDAYLAST>
          - <HOLIDAYFIRST>
              <xsl:value-of select="td[9]" />
            </HOLIDAYFIRST>
          - <HOLIDAYLAST>
              <xsl:value-of select="td[9]" />
            </HOLIDAYLAST>
          </ROW>
        </xsl:if>
      </xsl:for-each>
    </xsl:template>
  </xsl:stylesheet>
```

Figure 8 An XSL document for data extraction.

3.3 Data Checker

Data Checker checks and validates all the data extracted from the previous steps to further reduce the amount of errors to be stored in the database. The corresponding error cases to be detected and corrected have to be pre-set by the administrator. For example, the fare information should be a floating-point number. Once some non-numeric characters are found in the item extracted from the web pages as the fare information, the whole process will be terminated and alert messages will be displayed to the system administrator.

3.4 Data Exporter

After a series of data retrieval, extraction, and correction, the Data Exporter will store the data extracted from target Web pages into the system database. Oracle XML-SQL utility (XSU) and DOM are used for inserting XML data to the Oracle database. These utilities provide an interface for data storing. All the extracted routes information is stored into one single database table to support the subsequent bus trip planning. Other than "routes" and "stops" information, "district" information can also be extracted from the XML documents generated in previous steps and stores it into the database separately.

3.5 Management Tools

To manage the data extraction process, a set of management tools is developed. The data extraction rules, in the form of XSL documents, are at the moment hand-drafted by the staff with the help of a set of Web-based interfaces. Also, the history of the XSL documents created so far is stored in the database for later retrieval, which can further ease the rule creation process. With the help of the properly configured tools, the staff can then initiate the data extraction process by simply providing the set of route numbers to be extracted and the remaining steps will be started one by one automatically.

3.6 Information Integration for Trip Planning

3.6.1 Integrating Information

Different operators structure their routes information differently. For example, some operators may include bus arrival frequency in the bus routes information while some do not. e-Bus Planner has to integrate information from multiple sources with different structures and convert them into one common data structure in order to support *inter-operator* bus trip planning. In our case, the integration is done by aligning bus stops of different operators. In this project, simple keyword-based matching of the names of bus stops and major buildings is adopted.

3.6.2 Trip Planning

The Trip Planning Engine is designed to search for bus trips with unlimited number of bus route interchanges. After receiving the origin and destination of the trip from the user, the engine first identifies the bus stops that are close to both the origin and the destination, possibly owned by different operators. Again, simple keyword search is used. After locating the stops, the engine checks for all routes that passing through the located origin and destination. If there is a single route that connects both locations, the engine will stop searching for more paths and suggests the path to the user.

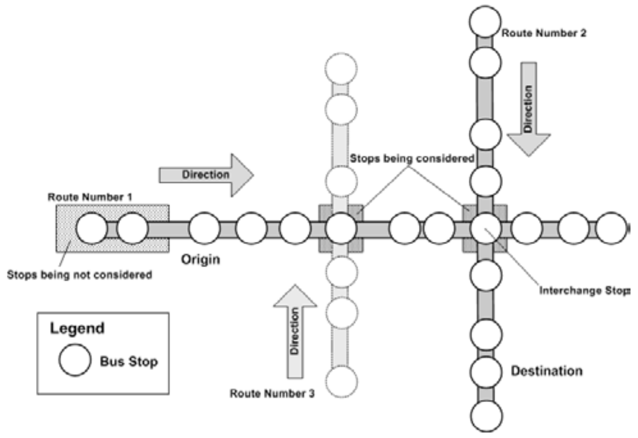
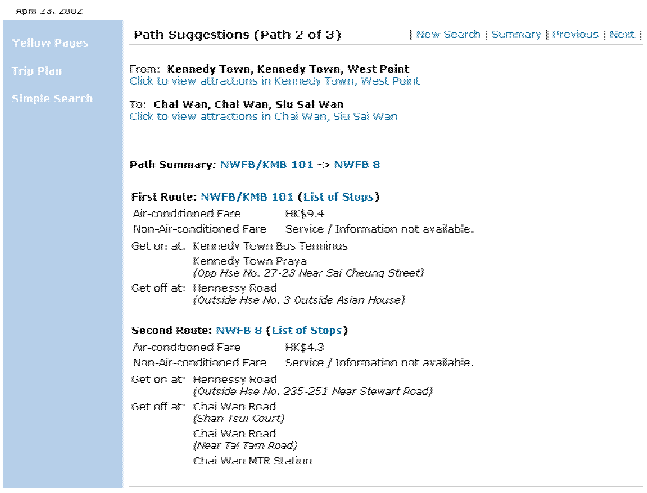


Figure 9 Path searching illustration.

As shown in Figure 9, if there is no single route providing direct bus services from the origin to the destination, the engine will start looking for paths by combining different routes. All the routes that go via the origin will be considered as the first route in the potential trip plans. All the stops of those routes that can interchange with other routes are identified as the potential interchange stops of the optimal trip plans. The engine then further checks for other routes that passing through the potential interchange stops as well as the destination. If at least one route is found, the engine will suggest the user to take the first bus and get on to that interchange stops then take the second bus to the destination. Of course, if still no paths can be found, the engine will go deeper into the path search tree in a similar manner until the destination is reached. The basic idea is similar to that of the breadth-first search.

3.7 A Prototype System

A prototype of e-Bus Planner has been developed and tested using the information published on the Web sites of two major public bus services operators in Hong Kong, namely the Kowloon Motor Bus and the New World First Bus. These two operators are operating about four hundreds bus routes covering most of the areas in Hong Kong. The latest version of e-Bus Planner has successfully extracted about more than a hundred of the bus routes from the corresponding Web sites for illustration. By providing the origin and destination to e-Bus Planner, plans of bus trips with a maximum of four different bus routes will be suggested to the user.



Path Suggestions (Path 2 of 3)

From: **Kennedy Town, Kennedy Town, West Point**
Click to view attractions in Kennedy Town, West Point

To: **Chai Wan, Chai Wan, Siu Sai Wan**
Click to view attractions in Chai Wan, Siu Sai Wan

Path Summary: NWFB/KMB 101 -> NWFB 8

First Route: NWFB/KMB 101 (List of Stops)

Air-conditioned Fare	HK$9.4
Non-Air-conditioned Fare	Service / Information not available.

Get on at: Kennedy Town Bus Terminus
Kennedy Town Praya
(Opp Hse No. 27-28 Near Sai Cheung Street)
Get off at: Hennessy Road
(Outside Hse No. 3 Outside Asian House)

Second Route: NWFB 8 (List of Stops)

Air-conditioned Fare	HK$4.3
Non-Air-conditioned Fare	Service / Information not available.

Get on at: Hennessy Road
(Outside Hse No. 235-251 Near Stewart Road)
Get off at: Chai Wan Road
(Shan Tsui Court)
Chai Wan Road
(Near Tai Tam Road)
Chai Wan MTR Station

Figure 10 A trip plan returned by e-Bus Planner involving two bus routes.

Figure 10 shows a screen shot of a bus trip plan returned by e-Bus Planner. It involves two bus routes operated by two different bus services operators. The trip information such as the route fare, the stops for getting off and getting on for route interchange are displayed to the user.

4. e-Food Marketplace

e-Technologies have long been known to be important to streamline processes in supply chain management. e-Food Marketplace is yet another project for illustrating the use of Web services as interfaces for making buyers' and suppliers' systems interoperable. To contrast with other related projects, we believe that the cost for Web services development and maintainance is still relatively high for small-to-medium enterprises (SME) and propose to build a case tool for wrapping buyers' and sellers' information systems with Web services. Building such a case tool is challenging as companies, even for those of the same industry, are likely to have their databases designed differently. To demonstrate the feasibility, we have implemented a prototype called *Web Services Builder* based on the java platform and the Java^{TM} Web Services Developer Pack (JWSDP). The tool enables the user to put semantic tags via a GUI to their internal data according to the ontology (knowledge representation) of a given industry. Then, the internal data schema is automatically mapped to a standard schema of the chosen industry. Based on the matching results, Web services code can automatically be generated. The current version of the prototype can assist the user to build interoperable Web services for product ordering, searching, price quoting and registration without writing a single line of code.

4.1 Web Services Builder - Conceptual Framework

In this section, the conceptual architecture of the case tool – Web Services Builder (WSB) is described in detail (see Figure 11). It basically consists of four important modules, each for a particular step in building Web services wrappers.

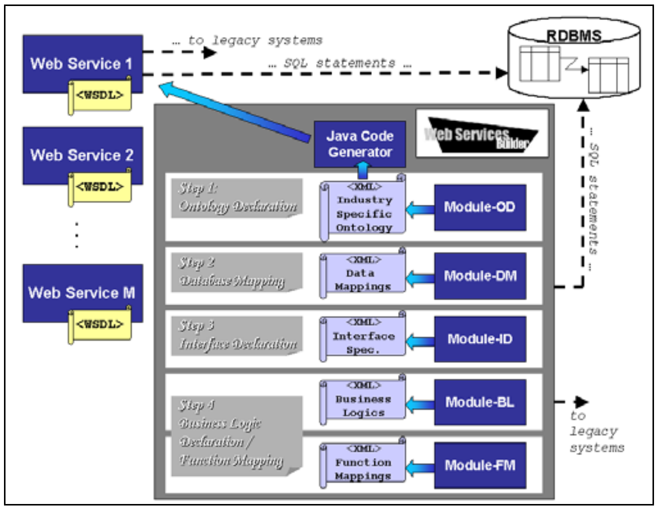


Figure 11 The system architecture of Web Services Builder.

Step 1 Industry-Specific Entity Declaration

In this project, the food industry is chosen. A small set of entities (by no means comprehensive) is adopted for illustration. Some important items are shown as follows:

Product is attributed by product id, name, price, stock level and description(s)

Order is attributed by order id, ordering date, ordering company, ordered product(s), ordered quantity

Registered Company (refers to the registered buyers) is attributed by company name, password, address, city, country, phone, email, contact person's name, phone and email.

It is possible to have a user interface for specifying different industry-specific ontologies, which however is not included in our current version of WSB.

Step 2 Database Mapping

Based on an agreed ontology, the subsequent step is to build the mappings between the ontology and the entities stored in the database. The different possibilities of the database design make this step not straight-forward to achieve. For instance, two possible ways to store products in the database are shown in Figure 12.

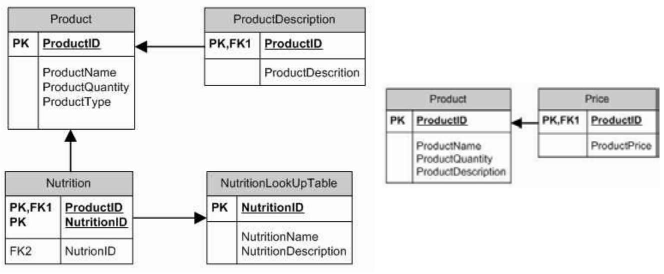


Figure 12 Two possible ways for representing products.

Our current version of WSB provides an interface to assist the users in first specifying the database connectivity related information (e.g., database driver name, connection string) and the database tables' foreign keys if not provided, and then creating the mappings between the database fields and the items in the ontology declared in Step 1 (i.e., product, order, company). The system robustness towards different schemas is currently achieved by checking with a list of pre-set schema patterns which are carefully designed manually. Figures 13-15 show the corresponding schemas used by WSB for storing the mappings' information which can in turn help generate the suitable SQL statements to select the right information from the database. To further reduce the mapping effort, automatic schema matching techniques can be explored [9].

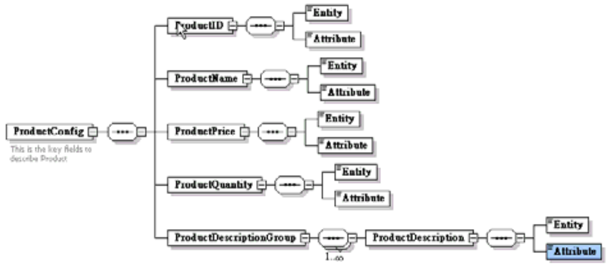


Figure 13 XML schema for product information mappings (visualized using XMLSPY).

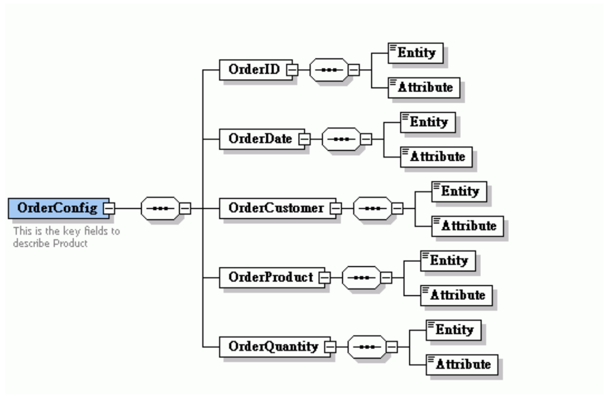


Figure 14 XML schema for order information mappings.

Furthermore, other related difficulties in database schema matching will also be encountered in this Step, including incompatibility in field's unit of measure, field's types, composite primary keys, etc.

Step 3 Interface Declaration

As we are not interested in providing a low-level database query functionality (as this will induce many security issues) but a set of on-line accessible services for enabling just-in-time execution of business processes in the marketplace, we need eventually the Web services interface declaration on top of the data-level mappings.

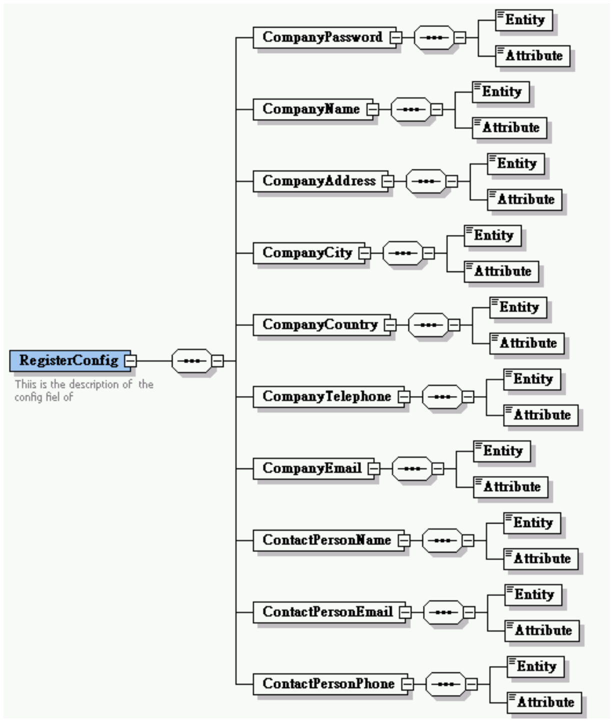
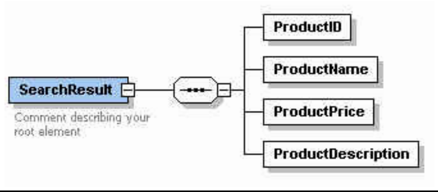


Figure 15 XML schema for registered company information mappings.

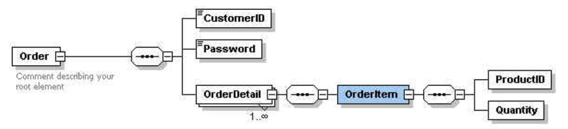
Here, we assume that the e-Food Marketplace with the help of WSB installation provides each participant with four different Web services, namely, product search, price quotation, registration (for potential buyers) and product ordering (password authentication needed). The specifications of their schemas as well as the WSDL document are listed below.

WS.1 Web service for product search

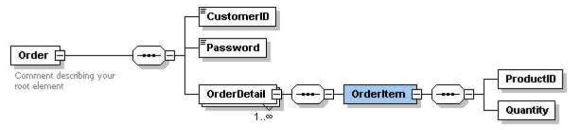
Input	Output
String: Keyword	String: Product described in the following XML format which can be extracted by the following XML schema.

Output Schema
 SearchResult Comment describing your root element ProductID ProductName ProductPrice ProductDescription

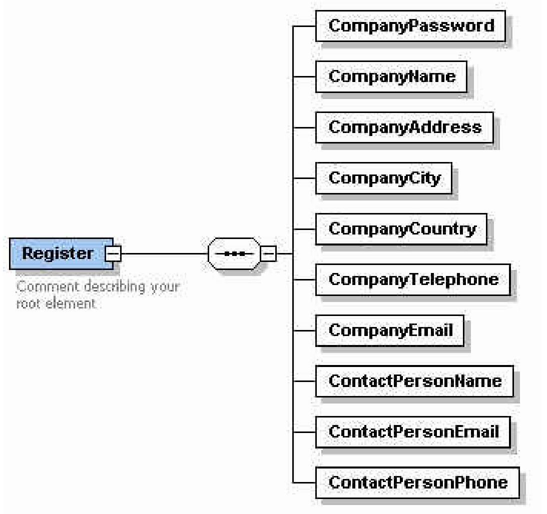
WS.2 Web service for product ordering

Input	Output
String: Order in XML format	String: Confirmation message with Order ID
Input Schema	
 Order Comment describing your root element CustomerID Password OrderDetail 1..∞ OrderItem ProductID Quantity	

WS.3 Web services for price quotation

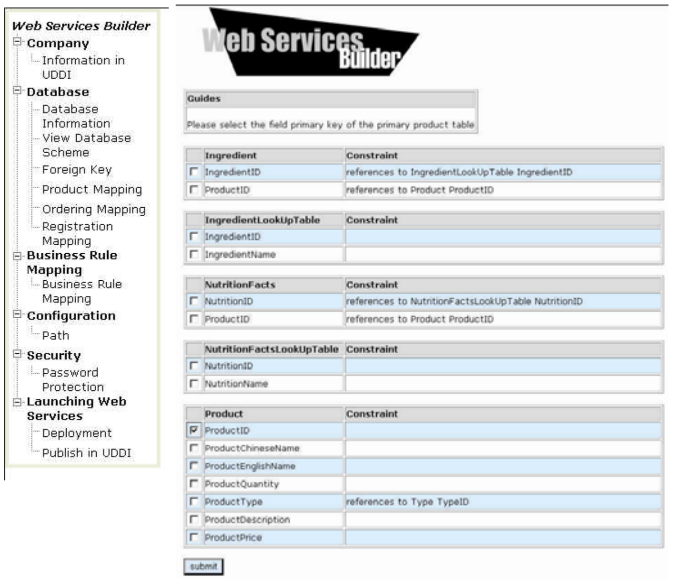
Input	Output
String: price quotation request described in XML format	String: price of the order
Input Schema	
 Order Comment describing your root element CustomerID Password OrderDetail 1..∞ OrderItem ProductID Quantity	

WS.4 Web service for potential buyer registration

Input	Output
String: Company information	String: Reject or accept messages with authentication information
Output Schema	
 Register Comment describing your root element CompanyPassword CompanyName CompanyAddress CompanyCity CompanyCountry CompanyTelephone CompanyEmail ContactPersonName ContactPersonEmail ContactPersonPhone	

Step 4 Business logic declaration / Function mapping

The Web services that can be created in Step 3 are all data-driven ones. In many cases, we need to incorporate business logics into the trading systems to partially automate some business processes. Incorporating these logics into the Web services wrappers without coding effort needs much more sophisticated tools. In our WSB, we only incorporate a discount delegation function in it. By properly specifying general and customer-specific discount rules, simple customized price quotation function can readily be supported and integrated into the price quotation Web service. Other than configuring some built-in functions for implementing business logics, another possibility is to explore the available APIs in the legacy internal systems and wrap them with Web services interfaces. However, such function mapping normally requires unavoidable coding effort.



Guides
Please select the field primary key of the primary product table

	Ingredient	Constraint
☐	IngredientID	references to IngredientLookUpTable IngredientID
☐	ProductID	references to Product ProductID

	IngredientLookUpTable	Constraint
☐	IngredientID	
☐	IngredientName	

	NutritionFacts	Constraint
☐	NutritionID	references to NutritionFactsLookUpTable NutritionID
☐	ProductID	references to Product ProductID

	NutritionFactsLookUpTable	Constraint
☐	NutritionID	
☐	NutritionName	

	Product	Constraint
☑	ProductID	
☐	ProductChineseName	
☐	ProductEnglishName	
☐	ProductQuantity	
☐	ProductType	references to Type TypeID
☐	ProductDescription	
☐	ProductPrice	

Figure 16 A snap shot of the Web Service Builder system.

After going through the four steps, four different Web services as well as their WSDL document will be generated automatically. The WSDL documents can then be published in a repository (e.g., UDDI) for later services discovery.

4.2 Enabled integrated services in e-Food Marketplace

Based on the interoperable Web services provided by the buyers/sellers registered in the marketplace, a number of value-added functions can be composed.

(A) On-demand and just-in-time price comparison

To purchase a particular product, one can discover related suppliers in the UDDI, look for those with the desired products and obtain just-in-time price quotations based on your existing business relationship with those suppliers as well as their stock levels. Other possible future extensions (available upon further enhancements of WSB) include product search under constraints (e.g., discovery time), autonomous negotiation and semantic service discovery using industry-specific ontology.

(B) One-stop e-Procurement and order-tracking support

Upon identifying some desired products to be purchased from different suppliers, the ordering requests can be sent to the corresponding suppliers and an integrated order-tracking can also be supported. Those order tracking information can later on be fed back into the autonomous negotiation module, if any, for optimizing future negotiation strategy.

5. Possible Extensions: Orchestration, Adaptation and Context-awareness – From An End-User Perspective

The aforementioned two projects are trying to demonstrate the readiness of e-technologies to provide value-added functions based on information integrated from heterogeneous sources. The first example that is related to public transportation could be a good starting point for an e-government initiative, which in turn could have an important impact on local tourism. The second example is a typical scenario requiring e-technologies for frictionless B2B e-commerce. They can both be extended to applications of larger scale and generalized to other problem domains without too many technical difficulties. Also, more sophisticated tools can be built to automate the manual processes (e.g., wrapper induction, schema matching). However, we believe that the impact of these information integration applications at their present forms is still limited. According to the value chain we described in Section 1, these projects have only reached at the middle part of the chain. We strongly believe that killer applications won't happen unless more value can be added up to a point where the end users can eventually find the applications not only powerful and resourceful, but also "familiar", "personalized" and "helpful".

For being "familiar", we propose that those value-added services should be integrated into some commonly used software platforms. In particular, we envision that one of the best places for e-technologies to reach people who works with computers would be their electronic or on-line diaries where they used to plan their events to be attended, clients to be met, order to be placed, tasks to be done, gift to be bought, etc. So, while using e-diaries, the users often need to make a lot of decisions, where the aforementioned value-added Web services can be tapped into as some enhanced functions, instead of treating them as another set of new systems with another set of interfaces to learn.

For being "personalized", we use the same scenario of e-diaries. E-diaries have a lot of personal information stored in it, e.g., phone books, users' past events and records, credit card information, etc. Those data provide an excellent source of the users' social networks where social intelligence related to the users could effectively be mined for decision support. For instance, if you put down "Dinner with Ann" on a particular day at a particular time, a "helpful" calendar function should be able to mine your relationship with Ann as far as possible based on your past schedules and consult some restaurants recommendation Web services with your as well as Ann's taste taken into account. Also, special promotions entitled by your subscribed credit cards could be a good source for providing smart suggestion. If the calendar function is an on-line version, knowledge about some preference patterns embedded in different users' schedules can in fact be discovered in a collaborative manner [10].

For being "helpful", we mean that the services directly provided to the end users should be of high-level and context-aware so as to address the users' need directly instead of expecting the users to go through some iterations to fine-tune them as far as possible. To achieve that, we need an easy way to orchestrate a set of publicly available Web services (some possibly interfaced with different sensors) for composing Web services at a higher level via declaration for problem solving. Problem Solver Markup Language (PSML), as suggested in [11], is proposed under a related scenario. Putting semantics in the service discovery mechanism with the use of OWL-S [12] is also important to further ease related composition tasks. Besides, helpful services should be autonomous as well as adaptive so as to identify the users' need and provide customized services proactively instead of staying passive. Good examples include

- finding the best way home under the current traffic and condition at around your usual time to leave office,
- prompting the users with special promotions which are of potential interest to their family members and available at this time in a nearby shop,
- negotiating with on-line retailers the prices of some possibly interested electronic products which are identified based on your interest profile as well as the products' recent sales, recent news about other related products to be released and user comments.

6. Conclusion

While integrating information from heterogeneous sources, as demonstrated by the two case studies, can provide us useful value-added functions, we argue that the end users need indeed services which are *familiar* in term of operation, *personalized* in term of how much the services know the user and his/her social network, and *helpful* in term of how much user intervention can be weaved so that high-level tasks can be dedicated to autonomous and adaptive composite Web services with context awareness. In order to meet the requirements of tomorrow's Web services-based systems (also called services-oriented computing), other than those recently proposed architectures and standards for Web-based system interoperability, we argue that methods for supporting social intelligence as well as distributed problem solving to be applied to the dynamic environment of the Web are two important areas that need much more research effort.

Acknowledgement

The author would like to thank Jonathan Leung and Spencer Lui for implementing the two prototypes described in the paper. This work is partially supported by Centre for E-Transformation Research, Hong Kong Baptist University.

References

[1] K.C. Laudon & C.G. Traver, *E-commerce: Business, Technology, Society*, Addison Wesley, 2002

[2] I. Muslea, Extraction patterns for information extraction tasks: A survey, in *Proceedings of AAAI-99 Workshop on Machine Learning for Information Extraction*, Orlando Florida, July, 1999

[3] M.E. Califf, Relational Learning Techniques for Natural Language Information Extraction. PhD thesis, Department of Computer Science, University of Texas, Austin, TX, 1998.

[4] M.P. Papazoglou and D. Georgakopoulos, Service-oriented computing, *Communications of the ACM*, October, 2003, Vol.45, No. 10, pp. 25-28.

[5] C. Peltz, Web Services Orchestration and Choreography, *IEEE Computer*, October 2003, Vol. 36, No.10, pp. 46-52.

[6] A. Pashtan, S. Kollipara and M. Pearce, Adapting Content for Wireless Web Services, *IEEE Computer*, October 2003, Vol. 36, No. 10, pp. 79-85.

[7] Myllymaki J. *Effective Web Data Extraction with Standard XML Technologies*. IBM Almaden Research Center, 2001.

[8] *XHTML 1.0: The Extensible HyperText Markup Language specification*. The World Wide Web Consortium (W3C). January 2000. http://www.w3.org/TR/xhtml1/

[9] E. Rahm and P.A. Bernstein, A survey of approaches to automatic schema matching, *The VLDB Journal*, 2001, Vol. 10, pp. 334-350

[10] W. Lin, S. Alvarez and C. Ruiz, Efficient adaptive-support association rule mining for recommender systems, Data Mining and Knowledge Discovery, 2002, Vol. 6, pp. 83-105.

[11] N. Zhong, J. Liu and Y. Yao, In Search of the Wisdom Web, IEEE Computer, November, 2002, pp. 27-30.

[12] OWL-S Home Page, http://www.daml.org/services