

# ebXML Registry and Repository Part 1: Business Domain

*ebXML Working Draft issue date July 28, 2000*

Version: 1.0

Project Team Lead: Scott Nieman, Norstan Consulting, USA

Editor: Joe Dalman, TIE Commerce, USA

Contributors: Scott Hinkelman, IBM, USA; Terry Allen, CommerceOne, USA; Lisa Carnahan, DOC/National Institute of Standards and Technology, USA; Eric Jarry, Software AG, France; Kunio Mizoguchi, Ecom Japan; Jong L Kim, Inno Digital Co., Ltd, Korea; Michael Rowley, Ph.D, eXcelon Corp, USA; Robert Miller, Global eXchange Services, USA; Sally Fuger, Ford Motor Company, USA; Yutaka Yoshida, Sun Microsystems, USA; Nagwa Abdelgfour, Sun Microsystems, USA; Gait Boxman, TIE Product Development, Holland

## *Abstract*

This document is Part 1 of the Functional Requirement Specification for an ebXML-compliant registry and repository. Part 1 defines the scope of the workflow to interact with a registry and repository, the actors involved in these interactions and the overall Domain Architecture for the e-Business Requirements of the registry and the repository. The Domain Architecture provides the basis of the detailed workflow specifications as defined in *ebXML Registry and Repository Part 2: e-Business Requirements* as well as forming the basis of traceability between Part 1 and Part 2.

## *Status of this Document*

This is the first public working draft of the *ebXML Registry and Repository Part 1: Business Domain*, issued by the ebXML Registry and Repository Project Team for public review and by members and project teams of ebXML.

This working draft incorporates the decisions of the ebXML Registry and Repository project team as of May 11, 2000. The first comment period opens May 15, 2000 and ends May 29, 2000. Comments with associated starting line number and ending line number, and proposed changes (required) should be emailed to [joe.dalman@tiecommerceusa.com](mailto:joe.dalman@tiecommerceusa.com) in DocBook, MS Word or plain text format. We will not accept this document edited with revisions on.

During the first comment period, the project team will review public comments while continuing its development of *ebXML Registry and Repository Part 2: e-Business Requirements*. Part 2 will focus on the detailed workflows that highlight the business-to-business (B2B) interchanges between users, a registry, and a repository.

## *Table of Contents*

ebXML Registry and Repository Part 1: Business Domain .....	1
ebXML Working Draft issue date May 11, 2000.....	1
Abstract .....	1
Status of this Document .....	1
Table of Contents .....	1
1 Introduction .....	2
1.1 Purpose .....	2
1.2 Dependencies on other Specifications .....	3
1.3 Conventions and Terminology.....	<b>Error! Bookmark not defined.</b>
2 Actor Relationships .....	4
2.1 Guest User .....	5
2.2 Corporation .....	5
2.3 Registration Authority .....	6
2.4 Industry Consortium .....	7
2.5 ebXML Business Application.....	8
3 Domain Use Case .....	9

53	3.1	Submitting Organization .....	10
54	3.2	Guest User .....	10
55	3.3	Independent Software Vendor (ISV) .....	11
56	3.4	Registration Authority (RA) .....	11
57	3.5	ebXML-compliant Business Application (ebAppl) .....	12
58	3.6	Semantic Mapping Specialist .....	13
59	4	ebXML Registry and Repository Architecture (Domain Package Diagram) .....	13
60	4.1	Interrelationships between <<UI>> .....	14
61	4.2	Interrelationships between <<UI>> and <<Services>> .....	15
62	4.3	Interrelationships between <<Applications>> and <<Services>> .....	16
63	4.4	Interrelationships between <<ebXML-Arch>> and <<Services>> .....	17
64	4.5	Interrelationships between <<Services>> .....	18
65	4.6	Optional Services within the <<Service>> Registry package .....	19
66	4.6.1	<<Service>> Transformation Service .....	19
67	4.6.2	<<Service>> Workflow Service .....	19
68	4.6.3	<<Service>> Quality Assurance Service .....	19
69	4.7	Optional Services within the <<Service>> Repository package .....	20
70	4.7.1	<<Service>> Library Control Service .....	20
71		Appendix .....	20
72			

## 73 1 Introduction

74 This *ebXML Registry and Repository Part I: Business Domain* working draft defines the scope  
75 and functional requirements of an ebXML-compliant registry and repository.

76  
77 Section 2 defines the actors that can interact with a registry and repository, and represents  
78 the relationships between these actors in an Actor Relationship diagram. This diagram is a  
79 hierarchical view that depicts actor inheritance and specialization as well as dependencies  
80 between actors.

81  
82 Section 3 defines the scope of the registry and repository as reflected by a high-level  
83 business domain Use Case. The use cases define the usage viewpoints from the perspective  
84 of several actors, including that of an ebXML business application.

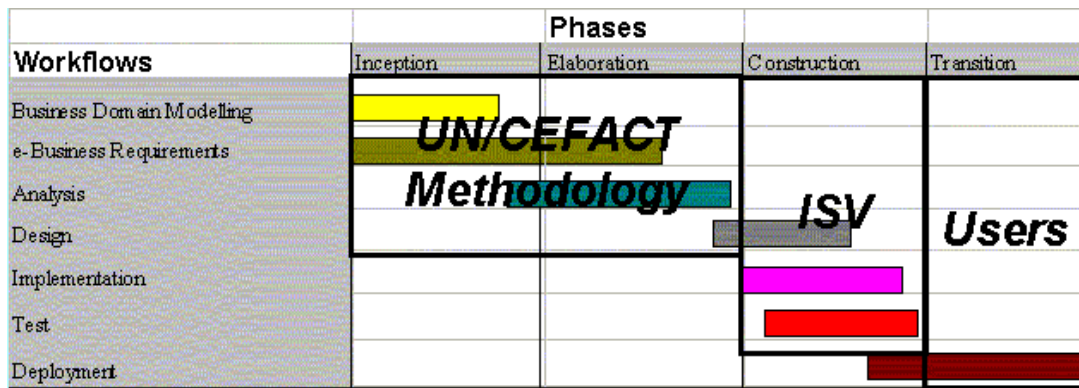
85  
86 Section 4 defines the ebXML Registry and Repository overall architecture through the use of  
87 a Domain Package diagram. Each package categorizes general functionality that could  
88 potentially become the software components, but not necessarily, since some functionality  
89 may be embedded within another service.

### 90 1.1 Purpose

91 The purpose of this working document is to define the overall system boundary as defined in  
92 the UN/CEFACT/TMWG UML Profile and Methodology. It is to define the scope of the overall  
93 system and provide a reasonable understanding of the size of the project.

94  
95 The methodology covers four main workflows as shown below, as this working document is  
96 the first of four documents.

97  
98  
99



**Figure 1. Phases and Workflows**

Ultimately, the result of the methodology will enable the ebXML Registry and Repository Project Team to apply Production Rules to generate the XML interface for both the Registry and the Repository. Software vendors can use the XML interfaces specifications as well as the UML models to understand the static and dynamic behavior of the system in order to construct the systems. With these specifications, conformance testing can be more clearly defined, since software components must conform to the XML interfaces, and the interactions to the system should be according to the UML activity and sequence diagrams.

It should be understood that this working document defines the scope for a network of registries and a network of repositories. This approach fits the model of world wide Internet, incorporating a vast number of interdependencies between companies, vertical domains relying on other vertical domains (finance, logistics, purchasing) and specifications relying on other specifications. It is envisioned that existing EDI directories could become repositories, and their traditional references to outside associations that maintain code lists (i.e., phone numbers, mailing addresses ) could be replaced by ebXML based links from one repository to another. While some organizations advocate the philosophy of a single global repository, ebXML Registry and Repository disagrees with this since the approach will not scale.

ISO specification 11179 defines a general model for registries and repositories and the roles and responsible parties with respect to those systems. This ISO specification has been used as a starting point for the ebXML Registry and Repository specifications.

## 1.2 Dependencies on other Specifications

Each ebXML Registry and Repository specification (Part 1 through Part 4) is based on the business requirements as documented in the ebXML Business Requirements Specification.

The Unified Modeling Language version 1.3, as issued by the Object Management Group, is used to define the entire ebXML Registry and Repository specification. The intent is a top-down model driven design approach versus a bottom-up approach based on data models or hand developed DTDs and brute force software development. This allows the business needs to drive the technology usage, provides a technology neutral representation of the specifications, and allows for the auto-generation of software code including XML specifications.

The UN/CEFACT/TMWG UML Profile and Methodology, which defines how and which UML artifacts need to be produced, is being utilized. It is adapted from the Rational Unified Process and specifically focused on e-Business Process modeling, including business-to-business (B2B) transactions. While its intent is to provide a protocol-neutral method of defining e-Business Processes, it also includes sample methods to auto-generate XML interfaces from the UML model by applying Production Rules. These XML interfaces follow a request/response model for real-time interaction with a registry and repository.

Regarding W3C specifications, ebXML Registry and Repository project team will consider any W3C Recommendations in the design or implementation of an ebXML-compliant registry and repository. W3C specifications in progress such as Xlink and Xpointer are key specifications that will be monitored. Such specifications should be helpful in enabling internetworking of registries to repositories and repositories to repositories.

Regarding non-W3C specifications, in the design and implementation, the ebXML Registry and Repository project team will consider any existing specifications that it finds of value to the project.

The Registry and Repository project team is considering transforming the Registry and Repository UML model to XML as according to the Object Management Group XML Metadata Interchange (XMI) v1.1 specification.

### 1.3 References

Many of the terms and definitions can be found in:

- ISO 11179
- OMG UML Specification
- TMWG Glossary
- OMG XMI Specification

They are not repeated here. The ebXML Registry and Repository project team will create it's own glossary which will be merged into the overall ebXML glossary.

## 2 Actor Relationships

The complete Actor Relationship class diagram depicts the hierarchical relationships, the unidirectional associations, and the dependencies between the actors. The actor relationships are not intended to specify role based authorization. The registration authority shall define the security policy.

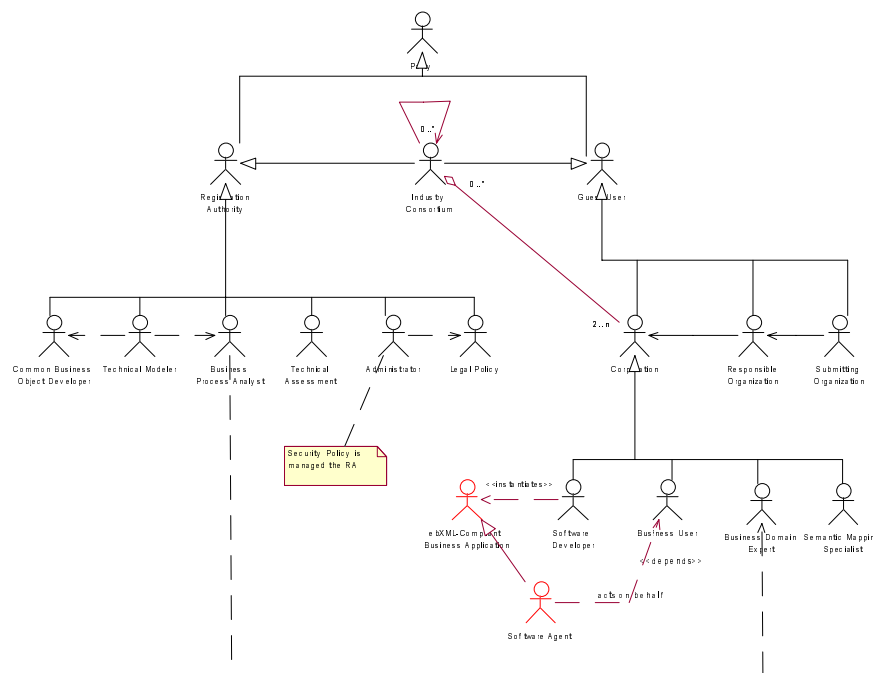
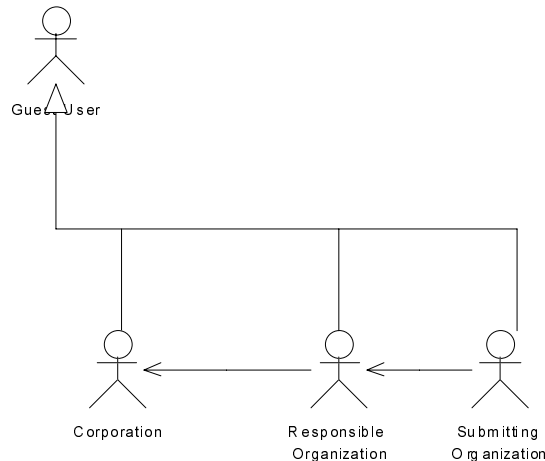


Figure 2.

175 The main areas in this hierarchy are elaborated upon below.

## 176 2.1 Guest User

177 The Guest User is an actor that inherits from the superclass actor Party. The Guest User is  
178 an organization or individual that acts on behalf of the organization, but has very limited  
179 responsibilities and privileges, e.g., access rights.



180

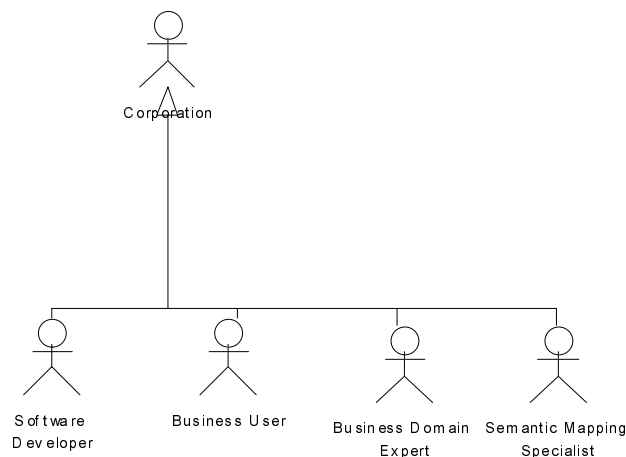
181 **Figure 3.**

182 Specialized actor classes of Guest User include the Corporation who is generally the end  
183 user of the registry and repository, the Responsible Organization and the Submitting  
184 Organization (for definitions of SO and RA, see ISO 11179). Unidirectional associations are  
185 drawn between these organizations; e.g., the Submitting Organization acts on behalf of the  
186 Responsible Organization who in turns acts on behalf of the Corporation.

## 187 2.2 Corporation

188 A Corporation is a collection of individuals that work together to achieve a common goal  
189 through its policies, processes, and resources. Shown are four types of actors, three of which  
190 could reside in a typical corporate business environment,; specifically, the Business Domain  
191 Expert, Business User and the Semantic Mapping Specialist.

192



193

194 **Figure 4.**

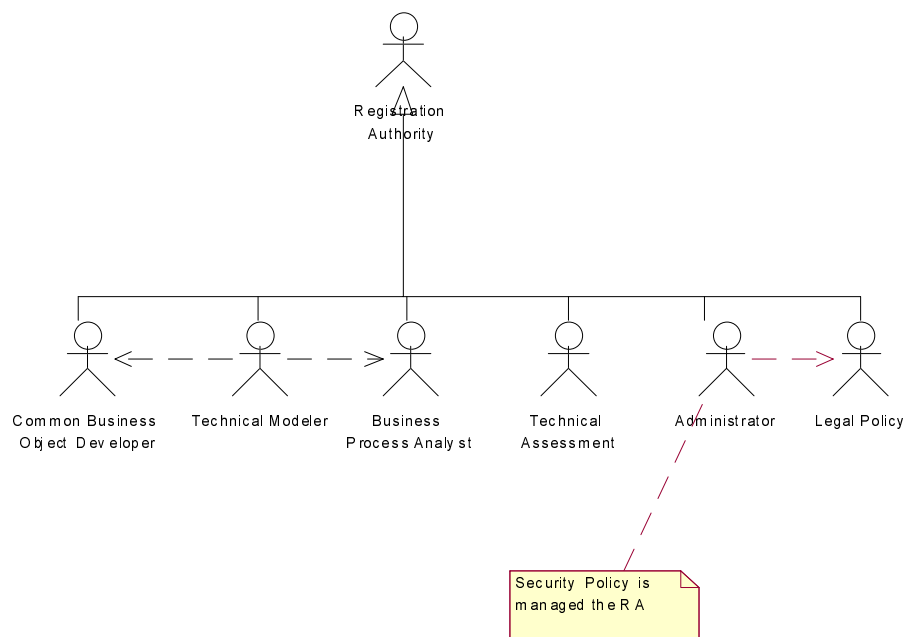
195 The Business Domain Expert is an actor that represents the corporate needs in the  
 196 development of B2B specifications. The Semantic Mapping Specialist is an actor that reviews  
 197 available B2B standards and specifications and maps them to the corporate internal business  
 198 processes and application metadata representations. The Semantic Mapping Specialist  
 199 traditionally supports EDI transactions or internal application-to-application (A2A) interfaces.

200  
 201 The Software Developer produces software that meets the needs of the Corporation, as  
 202 expressed by the Business Domain Expert.

## 204 2.3 Registration Authority

205 The Registration Authority (RA) is shown with several actors that could actually be part of the  
 206 RA itself, subcontracted resources, or in some cases volunteers.

207  
 208

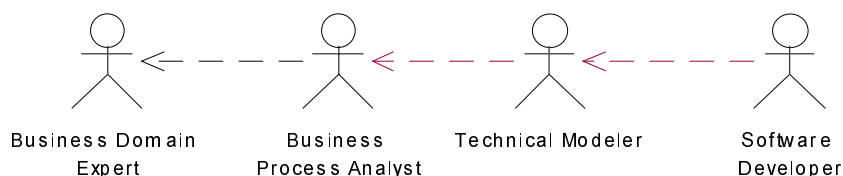


209

210 **Figure 5.**

211 Shown are six examples of roles within the Registration Authority. Three actors support the  
 212 knowledge extraction process to define the business and functional requirements of B2B  
 213 transactions: the Business Process Analyst, the Technical Modeler, and somewhat indirectly,  
 214 the Common Business Object Developer. These Actors are found in the UN/CEFACT/TMWG  
 215 UML Profile and Methodology. The Business Process Analyst is a facilitator with the skills to  
 216 ask questions of the Business Domain Expert and is vertical domain “neutral” in order to  
 217 provide alternative recommendations to the Business Domain Expert

218



219

220 **Figure 6.**

221 The Technical Modeler is familiar with the selected UML modeling tool and knows how to  
222 interact with the repository for model development and to search for existing patterns. The  
223 Technical Modeler is reliant on the Business Process Analyst skills in extracting knowledge  
224 from the Business Domain Expert.

225  
226 The Common Business Object Developer is skilled in reviewing UML models and detecting  
227 “patterns” in both business process models (activity diagrams and sequence diagrams) and  
228 data / design patterns (class diagrams). The Technical Modeler is also dependent on the  
229 Common Business Object Developer, who has the ability to obtain reusable common  
230 business objects for inclusion into the models in development.

231  
232 The Technical Assessment actor is responsible for UML model consistency and clarity. One  
233 task of the Technical Assessment actor is to identify semantic overlap, where equivalent  
234 semantic information is represented in different ways. Technical Assessment works with the  
235 Common Business Object Developer to resolve such issues. The Technical Assessment  
236 analyst is responsible for reviewing an industry submission of a model, identifying the  
237 submission’s relevance, uncovering any semantic overlaps with other specifications in the  
238 repository, assigning model integration project plans, and issuing recommendations for the  
239 final approval of a proposed specification. Model integration is a step that harmonizes  
240 multiple submitted models that are indirectly inter-dependent to each other. This occurs  
241 primarily between vertical domains, in which each vertical domain names one or more  
242 semantic units differently (ref: UN Layout Key).

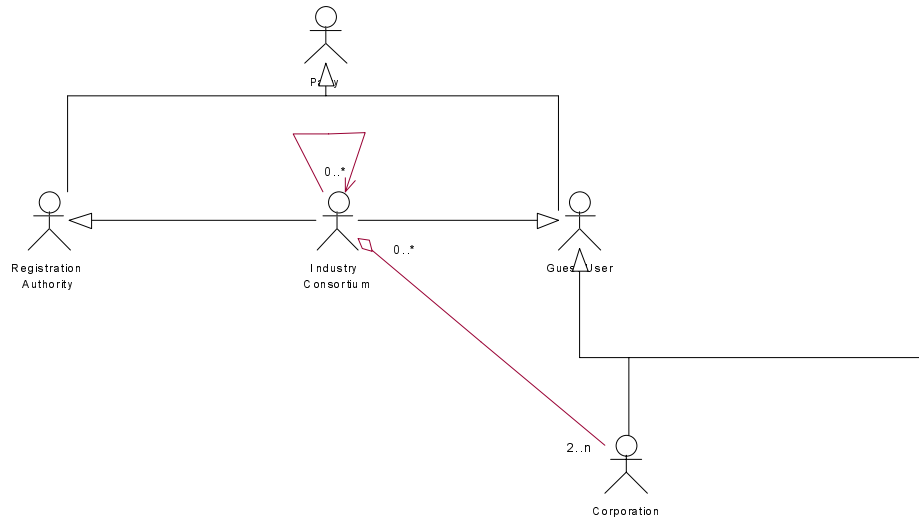
243  
244 The Administrator actor is responsible for the day-to-day operation of the ebXML Registry and  
245 of the Repository. This includes networking, server scalability, and usage statistics. The  
246 administrator has a thorough understanding of the overall *ebXML Architecture*, *ebXML*  
247 *Transport, Packaging, and Routing specifications*, and the APIs to the Registry and  
248 Repository as defined in *ebXML Registry and Repository Part 4: Design* (to track the number  
249 of API requests).

250  
251 The legal/policy actor is responsible for the business arrangements, legal policies, and  
252 operational policies of the registry and repository.

253 **2.4 Industry Consortium**

254 The Industry Consortium can be a Registration Authority as well as a Guest User therefore  
255 multiple inheritance is shown. In reality, the inheritance to Registration Authority has optional  
256 cardinality, as the Industry Consortium may not choose to host its own ebXML-compliant  
257 Registry and Repository.

258



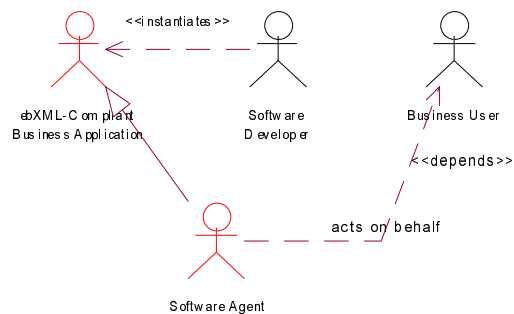
259  
260  
261

262 **Figure 7.**

263 The aggregation between the Industry Consortium and the Corporation shows that an Industry  
264 Consortium consists of two (2) or more Corporations. The Corporation may belong more than one  
265 Industry Consortium or none at all. This also shows that a corporation may act as a Registration  
266 Authority. The reflexive aggregation relationship from the Industry Consortium to itself illustrates that a  
267 consortium may have consortia members.

## 268 2.5 ebXML Business Application

269 An ebXML Business Application is an application that can communicate with the ebXML-  
270 compliant registry. An independent software vendor may develop such an application or it  
271 may be developed internally by a corporate development team using the ebXML  
272 specifications for a given problem domain. This is shown by the “**instantiates**” relationship  
273 between the Independent Software Vendor and the ebXML-compliant Business Application.  
274



275  
276

276 **Figure 8.**

277 A specialized actor class of the ebXML-compliant Business Application is the Software Agent.  
278 This is shown by an inheritance relationship to the ebXML-compliant Business Application  
279 actor. Software agents are based on artificial intelligence technology and are emerging in the  
280 industry at a rapid pace. A software agent acts on behalf of the Business User actor who  
281 configures the agent by specifying a unique profile that classifies a problem domain. This is  
282 noted by the “dependency” association between the Software Agent and the Business User.

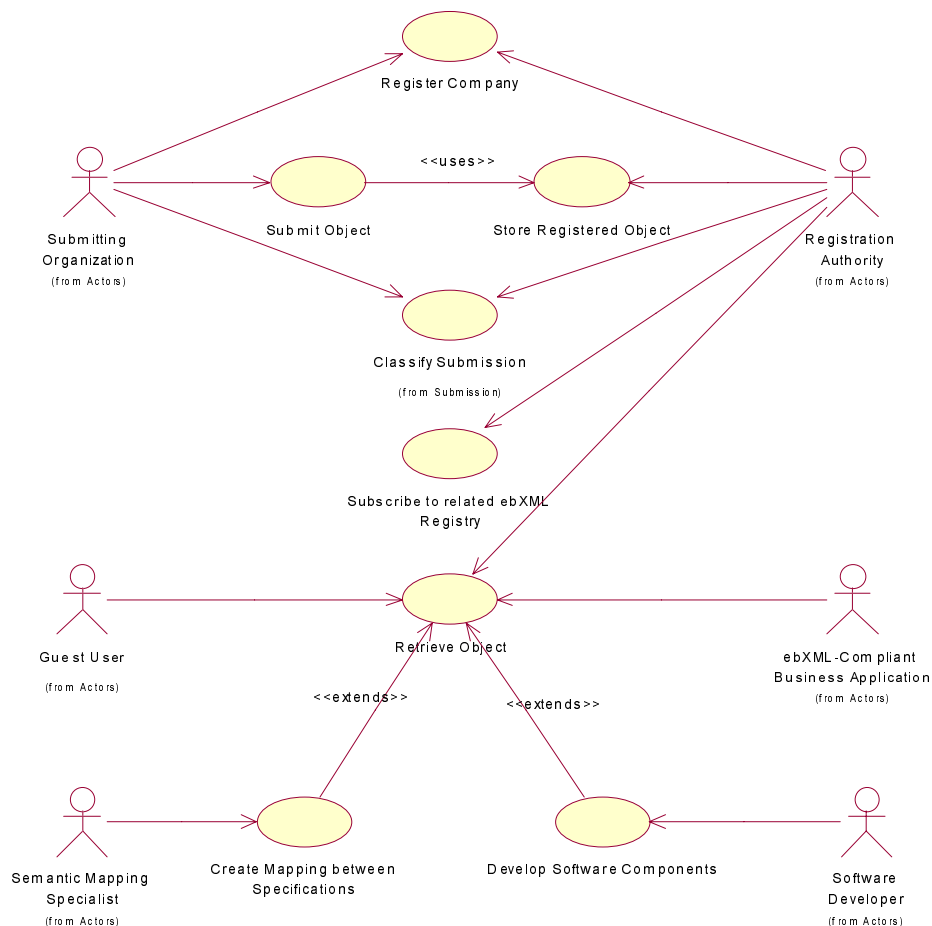


283 The unique behavior of the software agent is that it typically runs in an autonomous mode  
 284 without initiation by the Business User, and presents to the Business User information that  
 285 conforms to the profile. Software agents could monitor B2B transactions for anomalies,  
 286 detect trends, find new trading partners, and configure B2B integration systems.  
 287  
 288 Software agents will be able to interact with an ebXML-compliant registry, and request  
 289 specifications for e-Business Processes and any referenced data interchange specifications  
 290 in the e-Business Process.

### 291 3 Domain Use Case

292 The overall scope of the registry and repository is the submission, classification, and storage  
 293 of specifications and proposals, while allowing the creation of new specifications. Domain  
 294 Use Case provides an overall, high-level context of the business problem, a boundary to the  
 295 scope of the functionality, and traceability to the actors involved as defined in the Actor  
 296 Relationships hierarchical diagram. It does not delve into how the system performs the tasks  
 297 on a functional basis.

298  
 299 All actors are human actors with the exception of the ebXML Business Application. (It does  
 300 not appear as a different UML icon as suggested by other modeling conventions.) It is  
 301 important to refer back to the Actor Relationship diagram to understand the various  
 302 associations between the actors in the context of their roles. This approach enables a simple  
 303 Domain Use Case diagram as shown below.  
 304



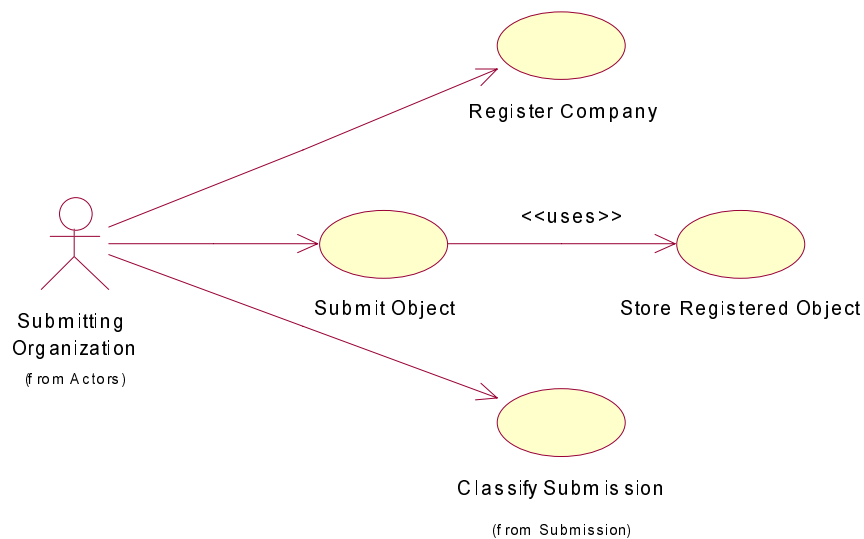
305  
 306 **Figure 9.**

307 Detailed discussion of each actor's viewpoint follows.

### 308 3.1 Submitting Organization

309 The Submitting Organization (SO) is responsible for the submission of items that are to be  
310 included into an ebXML Repository and referenced in an ebXML Registry. Depending on RA  
311 policies, the SO may first register itself and be approved to submit items as noted by the  
312 Register Company use case. After the RA approves the SO (see 3.4), appropriate access  
313 rights are granted that allow the SO to maintain its submissions and their metadata based on  
314 the different status codes specified by *ebXML Registry and Repository Part 3:Analysis*.

315  
316 The Submit Object use case provides the ability to submit a package. The Submit Object use  
317 case always "uses" the Store Registered Object use case to enter the submission into the  
318 repository. Since an object can contain other embedded objects, each object within the  
319 collection may also be registered in the registry, at the discretion of the SO. The RA may at a  
320 later date create another collection, and relate that collection to the SO collection.  
321



322

323 **Figure 10**

324 The SO may attempt to classify its submission, however, it is only a recommendation for the  
325 classification as the Registration Authority may change or add classification information after  
326 researching the submission. A record is created in the ebXML Registry (hence the term  
327 "register"), and a submission object is created in the repository as a container of the  
328 submission, which may be a collection of various documents, models, DTDs, schemas and  
329 other related information.  
330

### 331 3.2 Guest User

332 A Guest User is only granted read access. The Retrieve Object use case is abstract and  
333 includes the ability to directly retrieve data from the Repository or to search the ebXML  
334 Registry metadata in many ways, such as through unique identification, classification  
335 schemes, browsing, or complex search mechanisms. In the latter instance, a request may be  
336 sent from the Registry to the Repository to obtain a copy of the requested item.  
337

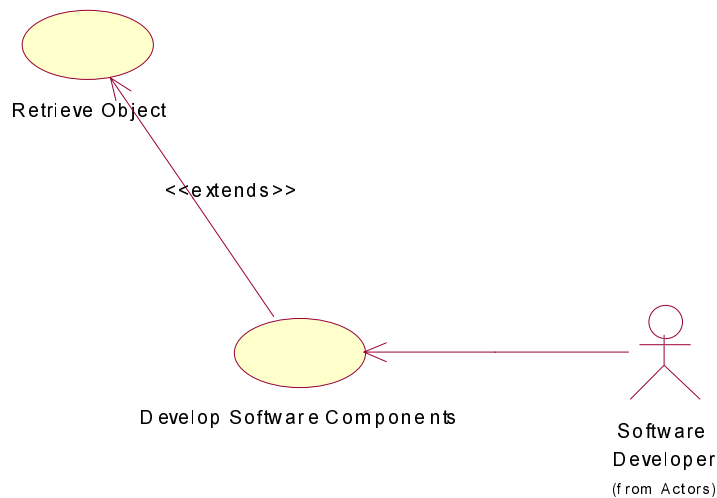


**Figure 11.**

There is an implied trusted security relationship between the Registry and Repository that should be transparent to the Guest User. Through the use of the Security Services and dependent on the access rights defined by the RA, the Guest User may need to authenticate to the Registry.

### 3.3 Software Developer

The Software Developer actor uses the ebXML Registry and Repository to retrieve B2B specifications, in order to Develop Software Components. The Develop Software Components use case is extended by the Retrieve Object use case since at some point the specifications must be downloaded or viewed on-line. It is not required at all times (optional) throughout the software development life cycle, therefore the <<extends>> stereotype is used versus the <<uses>> stereotype.



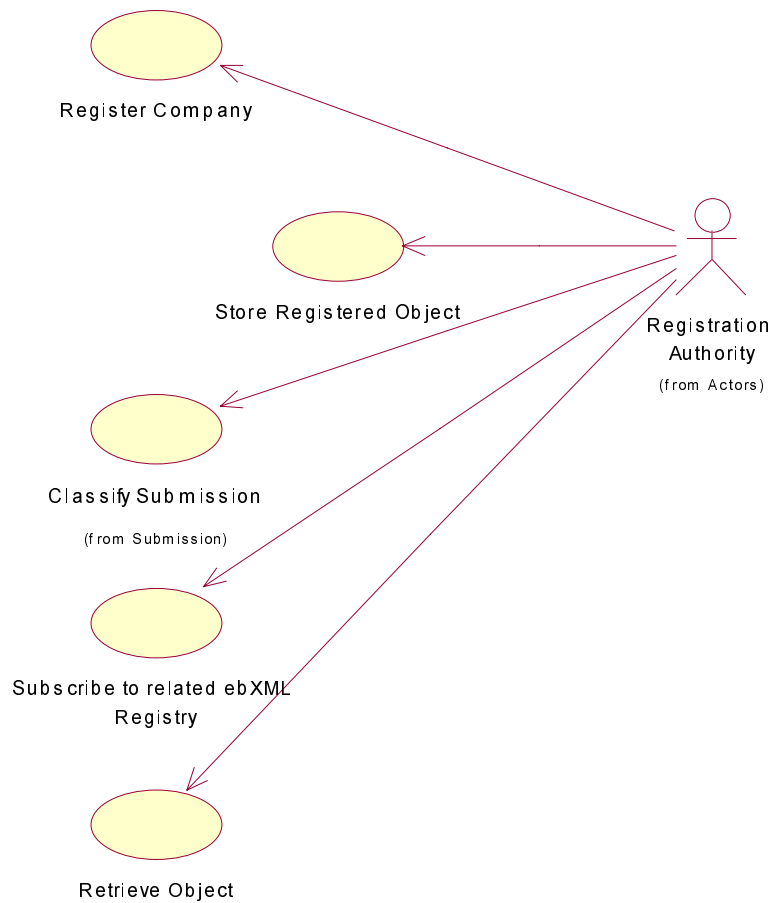
**Figure 12.**

The software developer can, through its association to the SO, register and submit software components that have been constructed, tested, and certified to be compliant to the retrieved specification(s). The components will then be available for download by the Business User.

In addition, since the Software Developer is a special type of Guest User, the Software Developer may browse the registry and repository in a read-only mode to determine whether there are any specifications that may be of interest. This allows the Software Developer to determine if the Develop Software Components use case should be entered.

### 3.4 Registration Authority (RA)

The Registration Authority (RA) is the owner of an ebXML registry and has “create” access to at least one Repository. The RA may authorize or reject the SO’s registration request that was initiated in the Register Company use case. Which action is taken by the RA is beyond the scope of the ebXML specification.



**Figure 13.**

The Store Registered Object use case allows the RA the ability to limit public access to any items that the SO may have submitted. A submitted object may reside in a queue until the Submitted Object is reviewed by the RA and the RA grants access. The RA may have a policy to certify the SO and allow the SO to store an object without prior review by the RA. Factors that may go into certification criteria include the number of submissions, accuracy of the SO's classifications, and ability for an software developer to produce software components based on the SO's specification quality.

The Subscribe to related ebXML Registry and Repository use case allows a Registration Authority to subscribe to an ebXML compliant Registry and Repository which may be of interest to the RA. For example, the RA may note that a submitted object is dependent on a specification that resides in another ebXML compliant Registry and Repository administered by separate RA. If that specification changes, the SO should be informed of that change and perhaps research that change to determine whether the change impacts the submitted object.

### 3.5 ebXML-compliant Business Application (ebAppl)

The ebXML-compliant Business Application imposes run-time requirements on the ebXML Registry and Repository, causing response time to become a key factor in the Retrieve Object use case. If the ebAppl is smart (software agent), the search and retrieval of items can be more complex, and collections of items could be returned. Therefore, collections of items are in scope for Retrieve Object use case.

391



392

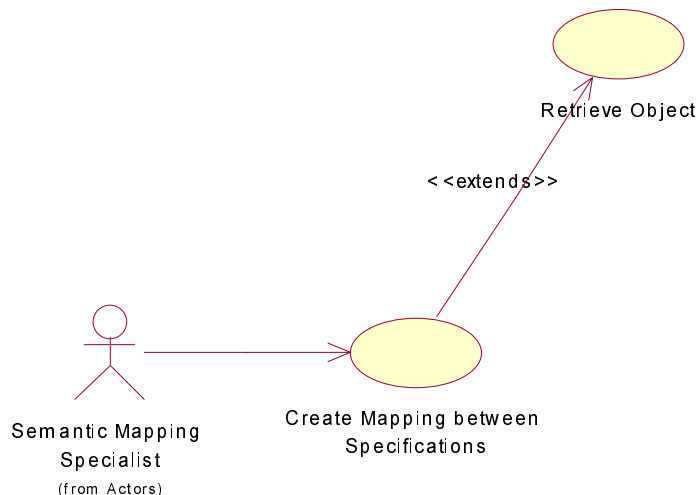
393

394 **Figure 14.**

395 An example of a reason for an ebXML-compliant Business Application to interact with an  
 396 ebXML Registry is to discover new trading partners that participate in a given marketplace, or  
 397 provide certain products and services. This type of data is supported by the ebXML Business  
 398 Process Metamodel, which is similar to eCo's Type Registry specifications.

### 399 *3.6 Semantic Mapping Specialist*

400 The usage of the ebXML Registry and Repository by the Semantic Mapping Specialist actor is  
 401 very similar in scope to that of the Software Developer, except that the creation of maps  
 402 includes the retrieval of two or more specifications. The interaction is the same in that the  
 403 Create Map between Specifications use case is extended by the Retrieve Object use case.  
 404 Specifications can be downloaded and mappings can be created off-line.



405

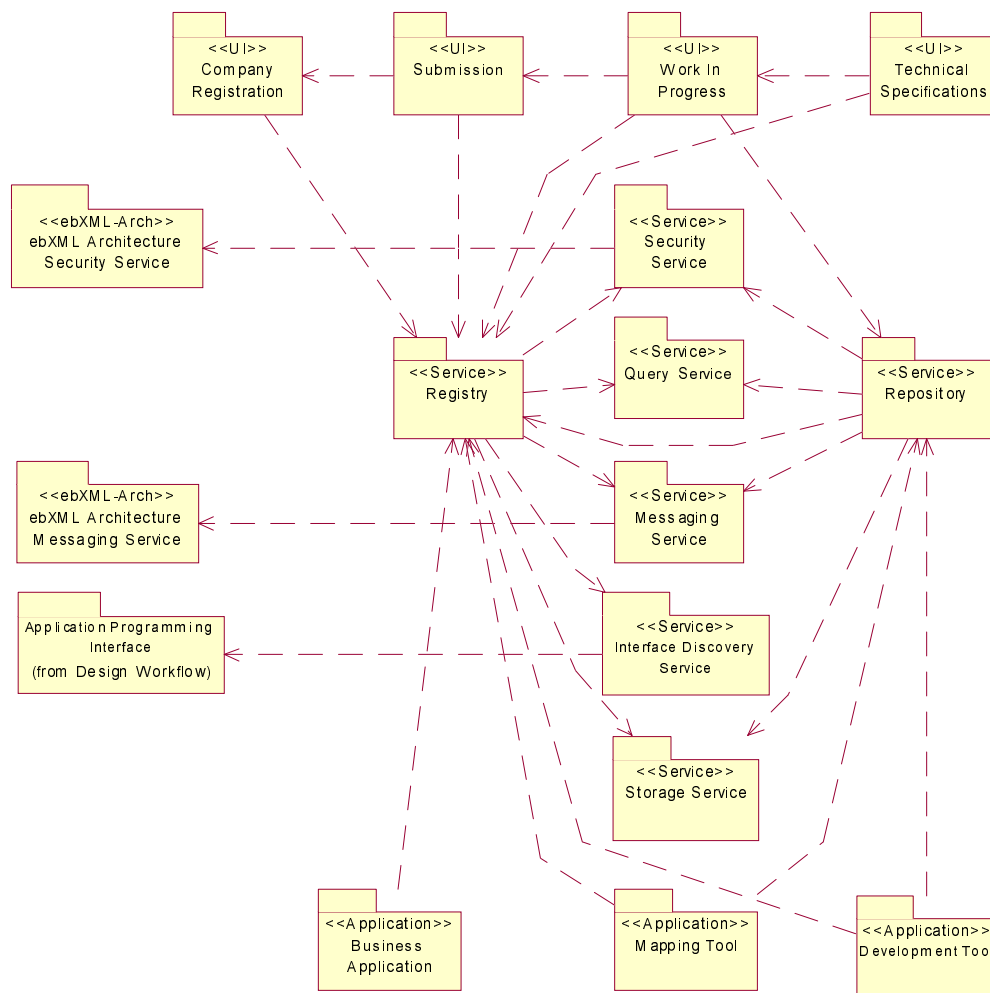
406 **Figure 15.**

407 The type of mappings could be "bridges" beyond two or more specifications, or integration  
 408 mappings to commercial software in which the vendor does not provide an integration  
 409 interface. Again, the Semantic Mapping Specialist is a special type of 1) Guest User for  
 410 read-only search and retrieval usage, and/or 2) Corporation which can submit its work  
 411 through its association to the SO. An ebXML compliant Registry and Repository will likely  
 412 contain mappings, likely in the form of XSLT.

### 413 *4 ebXML Registry and Repository Architecture (Domain Package Diagram)*

414 The following Domain Package Diagram details the various "subsystems" which carry out the  
 415 responsibilities for the domain use cases. A specific use case may be carried out by a  
 416 subsystem or an interaction between subsystems. The partitioning of the responsibilities for  
 417 each use case into packages or subsystems provides for an architecture that is intended to  
 418 scale with size and increased volume of usage over time.

419



**Figure 16.**

Within the Registry and Repository <<Service>> packages there are optional <<Service>> packages that cover additional capabilities. Each of these services will have its own API. The services are noted as optional or mandatory, and reflect whether a repository can be implemented by a file-based system, which is the minimum condition for implementation. These services are covered in section 4.6 and 4.7.

This document discusses the high level statement of the scope of each package and the interrelationships. The document *ebXML Registry and Repository Part 2: e-Business Requirements* provides more detail. Specifically, each package will have one use case diagram, and each use case will have a detailed activity diagram that describes the workflow.

#### 4.1 Interrelationships between <<UI>>

The <<UI>> stereotype provides the scope for the User Interfaces and the workflow required for companies and individuals to interact with the registry and retrieve information from the repository. There are four main areas that comprise the registry and repository as shown in figure 17.



**Figure 17.**

This shows how to submit an item or package of items to a repository, allow people to be aware that it has been submitted, and if desired, allow the submission to progress into an approved standard or specification. The Registration package is traceable to the Register Company use case. It will provide the use case diagram and detailed workflows for each use case in the diagram.

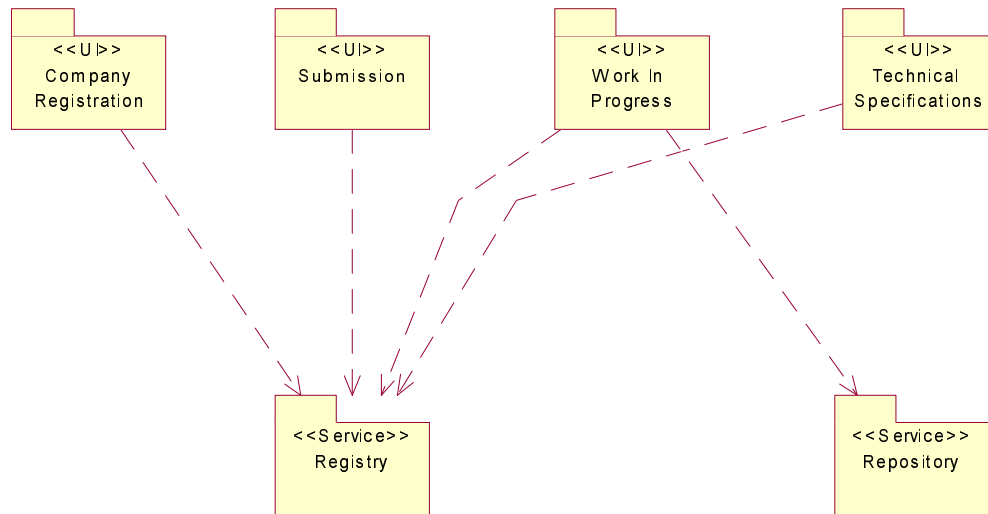
After an organization is registered with the RA and becomes an SO, it can submit items to the Registry and Repository. The Submission package is traceable to the Submit Items, Store Items, Classify Submissions, and Retrieve Items use cases. Unique classification schemes may be needed for each type of submission, since some submissions may have different important characteristics or attributes to describe them. Classifications will be detailed in *ebXML Registry and Repository Part 3: Analysis*, which will define the class diagrams for the registry.

If the SO intends for the submission to become an ebXML Specification to cover a business process, the submission may need to be enhanced. The Work In Progress package describes the workflow steps to become a Technical Specification, including library control functions such as checkin/checkout and versioning. Types of workflow steps may include enhancement of a submitted item to ensure that it meets the requirements of the ebXML Business Process Metamodel, harmonization with existing common business objects (Core Components) residing in the repository, and integration with other submitted items that may have overlapping semantics. Overlaps may occur for different vertical domains that have dependencies to each other. If a submission is not intended to be an ebXML Specification, but perhaps only a corporate specification, it still may reside in the repository but classified accordingly. In that case, there is no work to be performed, and the submission would pass rapidly through the Work In Progress package and flow into the Technical Specifications package.

The Technical Specifications package allows the submission to become a specification that can be viewed by **the public**. Work in progress is not available to the public unless in alpha or beta form, and classified as such. The Technical Specifications package provides the ability to browse the registry based on various classification schemes and to create simple and complex searches to find specifications.

#### 4.2 Interrelationships between <<UI>> and <<Services>>

In order for the <<UI>> packages to complete each step of the workflow, each package must interact with the Registry package. The Work in Progress package may interface directly with the Repository package to add modified copies when a library control service is provided with the repository. Since the registry maintains state information about the original submitted item, the original copy will always be retained.



482

483 **Figure 18.**

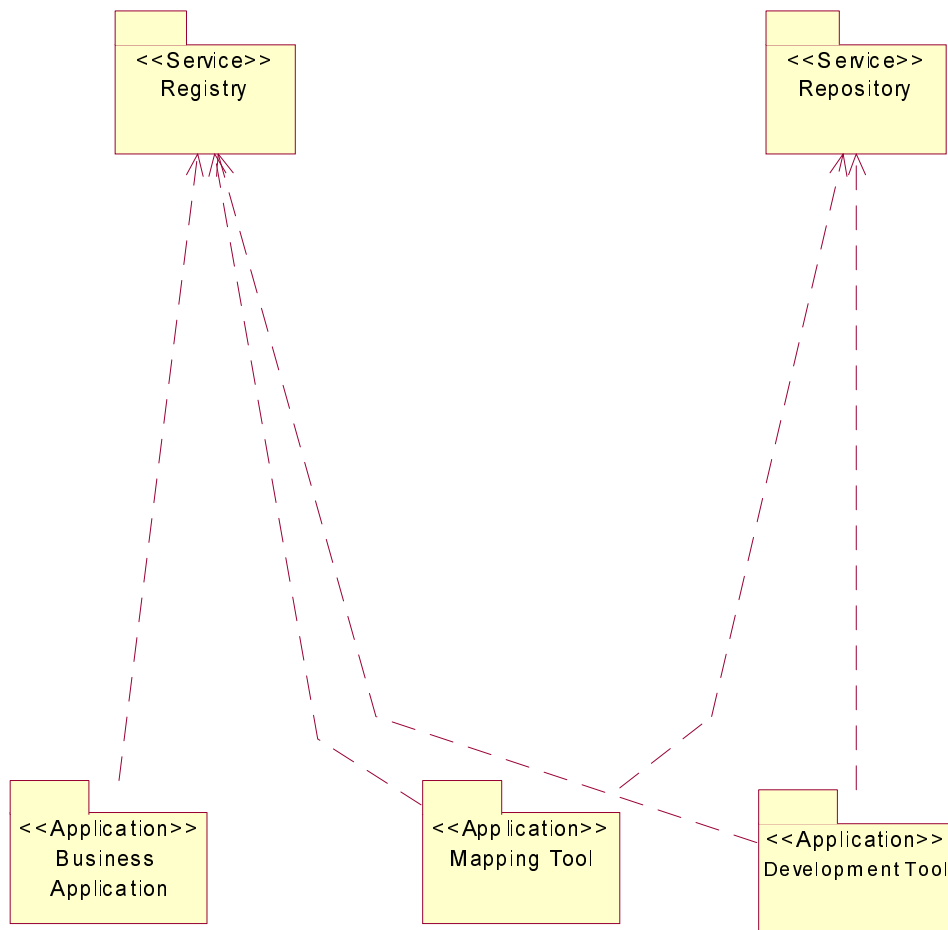
#### 484 *4.3 Interrelationships between <<Applications>> and <<Services>>*

485 The Registry and Repository also provide the ability for various development and business  
 486 applications to interact with them.

487

488 As in typical software development life cycle, application development tools interact directly  
 489 with the Repository. The two <<Application>> packages that are shown are the Mapping  
 490 Tool, and the Development Tool. While they are both development tools *per se*, the Mapping  
 491 Tools involves two or more specifications that need to be mapped at any one time. The  
 492 Development Tool is a typical Integrated Development Environment including commercial  
 493 Java, C++ and Smalltalk environments (VisualAge, Visual C++, Visual Café are examples).  
 494 These IDEs require various features of the repository including check-out/check-in services,  
 495 versioning services, transformation services for code generation as well as other features.





496

497 **Figure 19.**

498 The Business Application <<Application>> can only interface directly with the Registry, and  
 499 has no need for the repository services including check-out/check-in. This is traceable to the  
 500 Retrieve Item use case in which the ebXML-compliant Business Application interacts with that  
 501 use case.

502

#### 503 **4.4 Interrelationships between <<ebXML-Arch>> and <<Services>>**

504 The Security and the Message services are related to the overall ebXML Architecture for  
 505 security services and message services in conformance with the ebXML Transport, Routing,  
 506 and Package specification. Thus the authentication, certificate, and encryption techniques  
 507 are common across all ebXML-compliant Business Applications. Since the ebXML Registry  
 508 and Repository are also ebXML-compliant Business Applications, communication with these  
 509 services relies upon the ebXML TRP messaging services.

510



511

512

513  
514

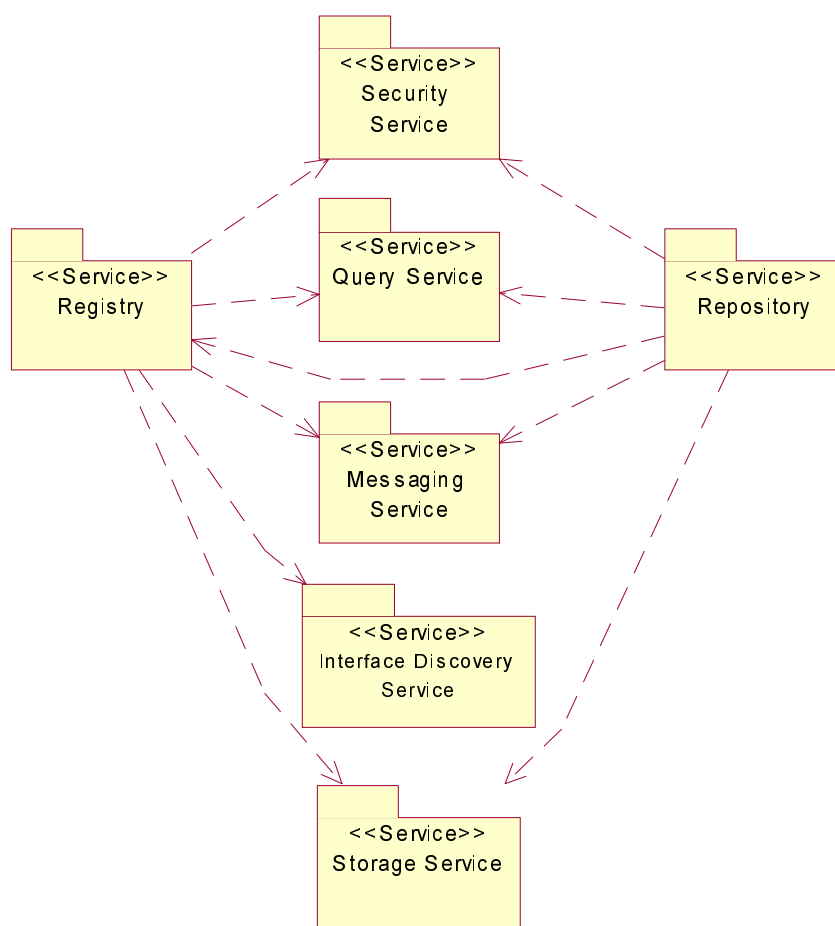


515 **Figure 20.**

516 **4.5 Interrelationships between <<Services>>**

517 As previously stated, the Registry and Repository both depend on a common security and  
518 messaging service, as formerly illustrated.

519  
520 For the simple case of retrieval of a registered item by reference to a unique identifier over  
521 HTTP it is desirable to support "THHTTP" as specified by RFC-2483 and revised by the OASIS  
522 registry and repository Technical Committee as well as TRP.



523

524 **Figure 21**

525 The dependencies between the Registry and Repository show a number of different features.  
526 First, the Registry's dependency on the Repository shows a publish and subscribe  
527 mechanism, in which the Registry receives metadata from a Repository particularly when a  
528 specification reaches a state in which the information needs to become public, or a new  
529 version is issued. The Registry may subscribe to more than one Repository besides its  
530 Primary Repository.

531 A Primary Repository is an ebXML-compliant Repository maintained by the RA to which the  
532 Registry has write authorization. This is shown by the Repository's dependency on the  
533 Registry to receive its content and establish a URI link to the content. This URI link must be  
534 maintained by the Registry.

535  
536 The dependencies on the query service package show that metadata in the registry can be  
537 searched to find information regarding contents of the repository. This metadata describes the  
538 contents in the repository including its URI for the ability to retrieve a copy of the item. In  
539 addition, based on the RA's policies or types of information stored, the repository content  
540 could be searched. The types of searches our defined in *ebXML Registry and Repository  
541 Part 3:Analysis*.

542  
543 The dependencies on the storage service shows that registered items in the repository must  
544 be stored. In addition, the classification schemes and metadata are stored in the registry.  
545 The physical storage mechanisms are not specified, but could be as simple as a file system.  
546

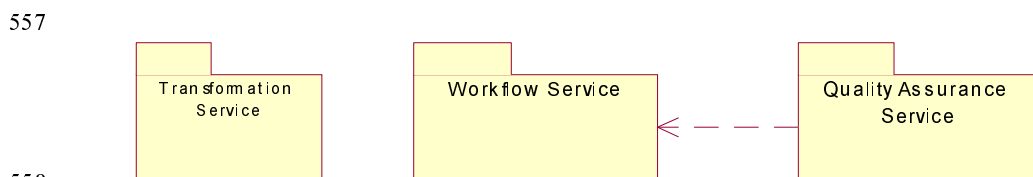


547  
548 Figure 22

549 The Interface Discovery service package describe a mandatory service that must be  
550 implemented by the RA. The RA chooses which optional services they will provide. Interface  
551 Discovery service is the mechanism that advertises which interfaces are available. Based on  
552 the security policies set up by the RA, these interfaces may be restricted.

553  
554 The application programming interface will be defined in *ebXML Registry and Repository Part  
555 4:Design*.

#### 556 4.6 Optional Services within the <<Service>> Registry package



558  
559 Figure 23.

##### 560 4.6.1 <<Service>> Transformation Service

561 The Transformation Service is used to transform objects into another form. (e.g., IDEF-1X to  
562 XML, XML to XML Schema). *This is going to be defined by ebXML Registry and Repository  
563 Part 2:e-Business Requirements.*

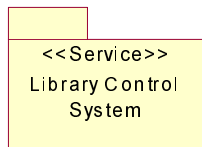
##### 564 4.6.2 <<Service>> Workflow Service

565 The Workflow Service carries out the workflows that are defined by *ebXML Registry and  
566 Repository Part 2:e-Business Requirements.*

##### 567 4.6.3 <<Service>> Quality Assurance Service

568 The Quality Assurance Service is used to validate content based on its classification and is  
569 only invoked on certain workflow events, dependent on the policies and procedures of the RA;  
570 e.g., registration of ebXML business process model. *This is going to be defined by ebXML  
571 Registry and Repository Part 2:e-Business Requirements.*

572 4.7 Optional Services within the <<Service>> Repository package



573

574 Figure 24

575 4.7.1 <<Service>> Library Control Service

576 The library control system package is an optional service that supports development, e.g., the  
577 work in progress package. *This is going to be defined by ebXML Registry and Repository*  
578 *Part 2:e-Business Requirements.*

579 Appendix

- 580
- OASIS Registry and Repository Technical Committee
  - UN/CEFACT UML Profile and Methodology
  - OMG Meta Object Facility (MOF)
  - OMG Unified Modeling Language version 1.3 includes UML, OCL and XMI (located on Rational's web site)
  - ISO 11179
- 581  
582  
583  
584  
585  
586

587

588

589

590