DETECTING THE EMERGENCE OF NEW CONCEPTS IN WEB COMMUNITIES

Pierluigi D'Amadio
Paola Velardi

Dipartimento di Informatica, via Salaria 113, Roma, Italy
{velardi ,damadio}@di.uniroma1.it

Abstract. This paper describes a methodology to detect the emergence (or the disappearance) of concepts through the observation of natural language communications (NLC). NLC are the documents, e-mails, written communications of any kind, that the members of a web community produce, access, and exchange for their purposes. The emergence of a new concept is suggested by the repetitive and consistent use of certain terms, while its intended meaning and appropriate conceptualization is obtained through a combination of text mining and algebraic methods.

1 The Self-Evolving Glossary

Building a glossary of terms is often the first step to model emerging knowledge domains and to favor interoperability between widely distributed communities of interest, who upload, exchange and share relevant information through the web. Modeling web communities in the IT society is significant for several reasons (Flake et al. 2002), that span from socio-cultural aims like the discovery of interdisciplinary connections, to more practical applications like the development of focused search engines, information filtering and information integration tools.

However, glossaries capture a static portion of a reality that can be instead highly dynamic, especially when modeling emerging domains. They are conceived and built as an "a priori" agreement on common terms, a "frozen" picture of the knowledge and competences of a community, that might suffer from a shortage of up-to date descriptions (Staab, 2002) (Heflin and Hendler 2000). On the other side, glossary building is a time consuming task, involving human effort to identify the relevant terms, agree on their meaning, and (in *thesaura*) structure terms according to some taxonomic ordering. In other terms, glossary creation is a consensus building process, often painful and tedious. There is an inherent risk in re-opening the process again and again.

The idea that we propose in this paper is that glossaries should be, as much as possible, *self-evolving*, continuously capturing the emergence of new concepts in dynamic web communities. The key to obtain this is *to simulate* the process of consensus building in humans, through a constant monitoring of natural language communications (NLC). NLC are the documents, e-mails, written communications of any kind, that the members of a web community produce, access, and exchange for their purposes. The emergence of a new concept is suggested by the repetitive and consistent use of certain terms in NLC. The simulation of consensus can be achieved through *statistical indicators*, aimed at selecting terms with certain distributional properties across the set of observed NLC.

This paper describes a methodology aimed at implementing the view of a self-evolving Glossary, detecting the emergence (or the disappearance) of concepts through the observation of natural language communications. Experiments have been made in several domains (art, tourism, web-learning, economy and finance), but in this paper we concentrate on an experiment related with the modeling of a web community organized through a Network of Excellence, INTEROP[1], on enterprise interoperability. Partners in INTEROP are academic and industrial institutions belonging to different research areas, grouped in three domains of expertise: Ontology, Enterprise Modeling, Architecture and Platforms. One of the main objectives of INTEROP is to model partner's competences in a Knowledge Map, indexed through a structured taxonomy of interoperability concepts. The KMap[2] aims at drawing a picture of the status of research in interoperability and to keep this picture up-to-date in the future. This provided us with an ideal test-bed for our methodology.

2 Collecting Evidences

The first step of the procedure is to collect a wide number of documents in written form, which should represent at best *what is communicated and exchanged* among the members of a community. This is a partly manual, partly automated step, and its complexity and involved effort strongly depends upon the community under consideration. For the purpose of the self-evolving Glossary, documents must be stored with an attached information about the source, authority and date of the acquired document. We have not developed a specific document warehouse architecture, since this depends upon the community document collection strategy and organization methods. In INTEROP, a collaborative platform in Zope/Plone has been adopted by the network partners (accessible from the INTEROP web site), which is also used to store documents and related metadata.

[1] http://interop-noe.org/

[2] details on the K-map can be found on the INTEROP web platform

3 Extraction of a Domain Lexicon

A *domain lexicon* L is a list of terms t commonly used within a given community of interest. The purpose of this phase is to automatically extract simple and multi-word expressions from the documentation collected in phase 1. Terminological *candidates* are multi-word strings with a precise syntactic structure (e.g: compounds, adjective+compound, etc) and certain distributional properties across the domain documents. Examples in various fields are the following: in enterprise interoperability: *enterprise intra organizational integration*, in tourism: *gourmet restaurant*, in computer networks: *packet switching protocol*, in art techniques: *chiaroscuro*. Statistical and natural language processing (NLP) tools are used for automatic extraction of terms (details are in (Navigli and Velardi, 2004)).

Statistical techniques are specifically aimed at simulating human consensus in accepting new domain terms. Only terms uniquely and consistently[3] found in domain-related documents, and not found in other domains used for contrast, are selected as candidates for the domain lexicon.

4 Extraction of Definitions

Once an initial lexicon is extracted, the subsequent phase is to obtain a list of (one or more) definitions for each term.

Extraction of definitions, as well as the subsequent step, which is glossary parsing, relies on a model of well-formed "definitory" sentences, that we describe through a set of *regular expressions*. Regular expressions, discussed later in a dedicated section, have several purposes:

- To *select* definitory sentences from those that are not. For example, many definitory sentences have the pattern "t is a Y", but using this pattern causes the extraction of a huge amount of non-definitory sentences, for example: *"Knowledge management is a contradiction in terms, being a hangover from an industrial era when control modes of thinking."* Regular expressions, along with statistical indicators, are used to prune this noise.
- To *prefer* definitory sentences with a precise structure often used by professional lexicographers, i.e. one that describes the meaning of a term by means of its kind (the so-called *genus,* or *hypernym*[4]) followed by a modifier (what *differentiates* the concept from its kind, the *differentia*). For example: "*Knowledge management is the systematic management of vital knowledge and its associated processes of creating, gathering, organizing, diffusion",* where the kind is *"systematic management".* A non-well

[3] Consistency of use across documents is measured through an entropy based measure called domain consensus

[4] In this paper *kind_of, genus* and *hypernym* will be used interchangeably to indicate the category to which a concept belongs.

formed definition, where no kind is provided, is: "*The core issue of knowledge management is to place knowledge under management remit to get value from it"* where no kind is explicitly provided.

- To *parse* definitory sentences in *order* to extract the *kind* information, and possibly more.

4.1 Extracting Definitions from Glossaries

Google recently provided a new search feature, called "*define*:" which can be used to search definitions of terms on web glossaries. However, using this search facility in an unconstrained way may cause the retrieval of a large number of often noisy (not pertinent to the domain) definitions. We defined the following algorithm to select pertinent definitions:

1) From the set of word components forming the extracted lexicon L of a domain D, learn a probabilistic model of the domain, i.e. assign a probability of occurrence to each word component. More precisely, let L be the lexicon of extracted terms, LT the set of word components appearing in L, and let

$$E(P(w)) = \frac{freq(w)}{\sum_{w_i \in LT} freq(w_i)}$$

be the estimated probability of w in D, where w∈LT and the frequencies are computed in L. For example, if L=[*distributed system integration, integration method*] then LT=[*distributed*, *system*, *integration*, *method*] and E(P(*integration*))=2/5

2) Search the terms in L using the Google "*define:"* feature. Select only those definitions def(t), t∈L, with the following features:

a) Domain pertinence: Let W_t be the set of words in def(t). Let $W'_t \subseteq W_t$ be the subset of words in def(t) belonging to LT. Compute:

$$weigh(def(t)) = \sum_{w \in W'_t, w \in LT} E(P(w))log(N_t/n_t^w)$$ where Nt is the number of definitions extracted for the term t, and n_t^w is the number of such definitions including the word w. The log factor, called *inverse document frequency* in the information retrieval literature, reduces the weight of words that have a very high probability of occurrence in any definition (e.g. *system*).

Definitions are ordered according to their weight. The first k definitions are selected, according to a threshold computed for each t[5]: $weigh(def(t)) \geq \vartheta_t$

[5] We omit the details for sake of brevity

b) Well formedness: apply a final filter to select those def(t) matching the "*genus-differentia*" style, expressed through a set regular expressions described in detail in section 2.3.

To compute the performance of this method in the worst ambiguity conditions, we selected 10 very ambiguous single-word terms in the INTEROP single word lexicon LT (including over 1000 words). Three evaluators marked the relevant and not relevant definitions (wrt the domain, i.e. enterprise interoperability). The inter-annotators agreement was 84%, since the task is inherently complex and subjective. We considered only the definitions marked in the same way by at least two annotators.

Table 1. Evaluation of definition selection algorithm.

Term	**R**	**A**	**Ra**	**N**	**N'**	**Pr=Ra/A**	**Rec=Ra/R**	**IAA**
Application	8	3	3	31	29	1.00	0.38	0.94
Component	4	2	1	28	26	0.50	0.25	0.93
Data	15	3	1	26	22	0.33	0.07	0.85
Design	5	1	1	39	36	1.00	0.20	0.92
Device	6	7	4	30	23	0.57	0.67	0.77
Framework	10	3	3	25	15	1.00	0.30	0.60
Knowledge	3	4	2	26	23	0.50	0.67	0.88
Process	8	2	2	38	33	1.00	0.25	0.87
Project	4	4	1	39	34	0.25	0.25	0.87
System	7	4	4	34	25	1.00	0.57	0.74
Average Performance after step 2a						0.71	0.36	0.84
Average Performance after step 2b						0.73	0.72	

Legenda: R=relevant definitions (majority-based), A=System-selected definitions N=extracted definitions, N' =definitions on which there is agreement (majority-based), Ra=R∩A, Pr=Precision, Rec=Recall, IAA=Inter Annotator Agreement.

Table 1 shows the results. Except for the last line, all numbers refer to the result of step 2a. The effect of step 2b (well-formedness) is a considerable improvement in recall, and a small increase in precision. Notice that the algorithm outputs always at least one of the relevant definitions, often the best, even though the annotators where requested to vote on a yes-no basis. Appendix I provides the complete output for the term *framework*. The definitions selected by the algorithm are underlined.

4.2 Extracting Definitions from NLC

As remarked in the introduction, the Dynamic Glossary needs continuous updates, as new terms and new fields emerge and are accepted within communities of interest. Definitions of new terms in well established communities and a new terminology in an emerging community are not found in glossaries, simply because of their novelty. But it is often the case that the inventors of these terms, or their initial users, provide a definition in their communications to the reference community. For example, the term "*federated ontology*" appeared only in 2001 in scientific literature (Stumme and Maedche 2001), but the first explicit definition is in a paper[6] dated 2004, that rephrases the concept of *federated ontology* proposed in a less explicit way in (Stumme and Maedche 2001) "*Federated ontologies are distributed, connected ontologies, somewhat analogous to federated databases*".

Identifying definitions in texts is much more complicated than choosing "good" definitions in glossaries. Definitions are buried in texts, and they cannot be recognized by means of simple regular expressions, like "X is a Y", since as remarked at the beginning of this section, these would produce an unacceptable amount of noise. We devised the following procedure:

Let L' be the list of terms in L for which no definition was found in the previous glossary search. For each t in L', do the following:

1) Extract from the community-provided documents first, and from the web after (only in case of unsuccessful search), a set of sentences including t. This implies some amount of pre-processing, like the treatment of various format, like *html*, *doc* and *pdf*. In case of web search, it is also necessary to handle limitations imposed by most search engines to multiple queries.
   A first filtering is applied, using regular expressions that match patterns like "*t is*" "*t defines*" "*t refers*" etc.
2) A second filter selects sentences which include, besides t, some of the words in LT (the set of word components appearing in L). The same probabilistic filter as in step 2a) of previous section is applied, with a small variation:

$$weigh(def(t)) = \sum_{w \in W'_t, w \in LT} E(P(w)) \log(N_t / n_t^w) + \alpha \sum_{w \in LT, w \in t} E(P(w))$$

   The additional sum in this formula assigns a higher weight to those sentences including some of the components of the term t to be defined, e.g. "*Schema integration* is [the process by which schemata from heterogeneous databases are conceptually integrated into a single cohesive schema.]"
3) Finally, the well-formedness criterion of previous section 2b is applied.

Terms are again selected according to a varying threshold, but, in this case, the threshold must be tuned for high recall, rather than high precision. In fact, for some terms, there might be very few definitions in literature and it is important to capture the majority of them.

[6] http://www.meteck.org/AspectsOntologyIntegration.pdf

Table 2. Evaluation of the definition extraction algorithm.

Term	R	A	Ra	N	Pr=Ra/A	Rec=Ra/R
application integration	5	6	3	35	0.50	0.60
collaborative system	2	11	2	16	0.18	1.00
distributed object technology	4	10	4	12	0.40	1.00
knowledge sharing	9	9	5	38	0.56	0.56
message exchange	2	3	2	20	0.67	1.00
ontology alignment	3	3	1	16	0.33	0.33
open standard	5	14	5	19	0.36	1.00
process integration	12	4	3	39	0.75	0.25
schema integration	10	4	1	30	0.25	0.10
service center	2	18	2	40	0.11	1.00
Average Performance (all steps)					0.41	0.68

Table 2 shows the performance obtained when searching 10 terms from the lexicon L. Appendix I (part 2) shows the definitions, with rating, extracted for the term: *ontology alignment*, a relatively new term in the area of ontology building.

After this phase of the ontology updating process, selected definitions are presented to domain experts with and indication of the source (document or web glossary) and authoritativeness. Experts can modify, reject or accept each definition[7].

5. Parsing of Definitions

This section adds further details on the definition and use of regular expressions. We use regular expressions[8] to select well-formed sentences and to extract kind-of relations from natural language definitions. The components of a regular expression are fixed words or word sequences, part of speech and syntactic chunks.

At first, sentence *chunks* (e.g. noun phrases NP, prepositional phrases PP, etc.) are identified using an available syntactic parser, the TreeTagger[9]. For example, the following regular expression is used to verify the well formedness criterion:

[7] In INTEROP an initial glossary relative to educational objectives has been acquired and evaluated. The interested reader might access on the web site the deliverable 10.1 to learn the details of this process. A second, large scale (1800 terms) interoperability glossary has been acquired and will be fully evaluated by the end of year 2 of the project.

[8] http://www.oreilly.com/catalog/regex/chapter/ch04.html

[9] TreeTagger is available at

r = "^(PP)?(NP)+"

This regular expression (see subsequent examples) prescribes a sentence structure at the chunk level: a definitory sentence is formed by a facultative prepositional phrase (^(PP)?) followed by the *main noun phrase* (NP), followed by anything else (+).

When a sentence matches the well formedness and probabilistic criteria described in previous section, other regular expressions are applied to extract additional information.

For example, the following regular expression at the word level is applied (with others) on the main NP to separate candidate definitions from non-definitions in step 1 of section 2.3.2:

p1="^(Refers|Referring)\\sto\\s(((a|the)\\s)?(type|kind)\\sof\\s)?(.*)" If a sentence is selected as being a definition, additional regular expressions are used to extract from the main NP the *kind_of* (*hypernym)* information.

For example, consider the regular expression

r1 = "^(A|D)?((V|C|,|J|N|R)*)(N)".

Symbols in r1 are part of speech tags (POS), e.g. article (A), verb (V), adjective (J), etc.

A sentence matching both **r** and **r1** is:

domain model: "*In the traditional software engineering perspective, a precise representation of specification and implementation concepts that define a class of existing systems.*"

When parsing with the TreeTagger we obtain:

Syntactic Chunks: (PP **NP** PP CNP RVP NP PP)

POS: (PAJNNN AJ**N** PNCNNWVANPJN)

The application of **r1** returns:

hypernym: representation

The bold POS (**N**) represents the fragment selected as the hypernym.

We then learn that:

$$\text{domain} - \text{model} \xrightarrow{kind-of} \text{representation}$$

Appendix I highlights in bold the hypernym extracted from selected definitions.

Table 3 shows the performances in three domains.

Table 3. Precision and recall of the hypernymy extraction task in three domains.

	Art	Interoperability	Computer Networks
Precision	0.973	0.947	0.955
Recall	0.957	0.914	0.932

http://www.ims.uni-stuttgart.de/projekte/corplex/TreeTagger/DecisionTreeTagger.html We augmented TreeTagger with regular expressions that capture named entities of locations, organizations, products, persons, and time expressions. This allows us to capture other relations besides hypernymy, but this research is still in progress.

6. Creation of a Taxonomy

Parsing definitions allows it to structure the terms in T in taxonomic order. However, ordering terms according to the hypernyms extracted from definitions has well-known drawbacks. An interesting paper (Ide and Véronis, 1993) provides an analysis of typical problems found when attempting to extract (manually or automatically) hypernymy relations from natural language definitions, e.g. attachments too high in the hierarchy, unclear choices for more general terms, or-conjoined hypernyms, absence of hypernym, circularity, etc. These problems are more or less evident – especially overgenerality – when analysing the term trees forest generated on the basis of glossary parsing.

To reduce these problems, we proceeded as follows:

1) First, we arrange the terms in T taxonomically according to simple *string inclusion*. String inclusion is a very reliable indicator of a taxonomic relation, though it does not capture all possible relations. This step produces a forest of sub-trees.
2) Then, we use hypernymy information extracted from definitions to capture additional taxonomic relations between terms *at the same level of generality* (e.g. in the example above: *representation, model, schema, ontology, knowledge data, information*).
3) If terms have more than one selected definition, or have or-conjoined heads in the main NP, more than one hypernym is extracted by the algorithm of section 2.3. However, we select only hypernyms belonging to the set of domain relevant words LT. Hence for example, *knowledge* has the following hypernyms: *information*, *fact-and-relationship* and *meaning*. Only the first is selected.
4) After step 3, component terms of the sub-trees ST_i have one or more hypernym associated. Given a term t: $t_l t_r$ (where t_l and t_r are left and right components of t, e.g. t=*enterprise application integration*, t_l =*enterprise application*, t_r =*integration*) we verify whether there is a multi-word term t' : $t'_l t'_r$ in the taxonomy such that $t_r = t'_r$ and either $t_l' \xrightarrow{kind\_of} t_l$ or $t_l \xrightarrow{kind\_of} t_l'$ (e.g. if t=*service integration* and t'=*application integration*, it holds that $service \xrightarrow{kind\_of} application$, and therefore $service\_\mathrm{int}\,egration \xrightarrow{kind\_of} application\_\mathrm{int}\,egration$).

Appendix II shows a small fragment of the complete INTEROP taxonomy[10] (the sub-trees rooted in *integration*) At the end of Appendix II we also show an excerpt of the detected hypernymy relations, used in step 4.

Ordering terms taxonomically is a highly subjective task, therefore is not easy to evaluate the output of this phase. Golden standard are not available, especially in sub-domains. However, we did a small experiment: given the initial *integration*, *interoperability* and *system* taxonomy, our method was able to detect 25 hypernymy relations, e.g.

[10] the taxonomy includes 1800 terms belonging to the three main domains of INTEROP, e.g. ontology, enterprise modeling, architectures and platforms.

$$schema \xrightarrow{kind\_of} design \xrightarrow{kind\_of} model \xrightarrow{kind\_of} representation$$

We compared these relations with the WordNet[11] general purpose lexicalised ontology, in the following way:

let $kind\_of(w_i, w_j)$ be a detected hypernymy relations between w_i and w_j, either a direct relation or a chain of hypernymy links, as in the *schema* example above.

If in WordNet it holds that: $kind\_of(S_i, S_j),\ Si \in (senses\ of\ w_i)\ Sj \in (senses\ of\ wj)$, where again *kind_of* is either a direct relation or a chain, then mark $kind\_of(w_i, w_j)$ as positive. For example, in WordNet there is a direct hyperonymy relation between sense #1 of *schema* and sense#1 of *representation.*

The evaluation showed that there are around 33% matches with respect to a “golden standard” taxonomy like WordNet, but on the other side, WordNet is a general purpose ontology, and some of the not-corresponding relations detected by our methodology seem still very reasonable in the interoperability domain, as the reader may verify evaluating the detected kind_of links in Appendix II. Notice that, as expected, the major problem is the over-generality of certain hypernymy links (e.g. everything is a “system”).

In any case, our purpose here is not to fully overcome problems that are inherent with the conceptually complex task of building a domain concept hierarchy. At the end of this process we obtain, a forest of trees where nodes (the concepts) are named as the corresponding terms in natural language, and the only semantic relation is hypernymy, even though ongoing research for extracting additional relations is progressing. Discrepancies and inconsistencies can be corrected by a team of human specialists, who will verify and rearrange the nodes of the sub-tree forest.

Acknowledgements

This work has been supported by the INTEROP network of Excellence IST-2003-508011.

[11] http://www.wordnet.princeton.edu WordNet is the most widely used and cited lexicalized computational ontology

References

(Heflin and Hendler, 2000) Heflin, J. and Hendler, J. *Dynamic Ontologies on the Web.* In: Proceedings of the Seventeenth National Conference on Artificial Intelligence (AAAI-2000).

(Kleinberg 1998) Kleinberg, J. *Authoritative sources in a hyperlinked environment*. Proc. 9th ACM-SIAM Symposium on Discrete Algorithms, 1998.

(Navigli and Velardi, 2004) Navigli, R. & Velardi, P. (2004). *Learning Domain Ontologies from Document Warehouses and Dedicated Web Sites*. Computational Linguistics, MIT press, (50)2.

(Staab, 2002) S. Staab, *Emergent Semantics*, IEEE Intelligent Systems, v.17 n.1, p.78-86, January 2002

(Stumme and Maedche 2001) 4. G Stumme, A Maedche, *Ontology Merging for Federated Ontologies on the Semantic Web*, Workshop on Ontologies and Information Sharing, IJCAI

Appendix I: Selection of definitions from web (1) and document warehouses (2)

Example 1: selection of appropriate definitions from glossaries: "**framework**" (selected sentences underlined, selected hypernym in bold)

Def: A **grid structure** where the vertical boxes depict the workflow of core processes, and the horizontal boxes depict business subsystems that control the lifecycles of key business objects
Weight : 0.1444115
Def: a **template** containing a sequenced set of all groups/segments which relate to a functional business area (or multi-functional business area) and applying to all messages defined for that area (or areas)
Weight : 0.12572457
Def: A body of **software** designed for high reuse, with specific plugpoints for the functionality required for a particular system
Weight : 0.10959378
Def: A framework is an extensible structure for describing a set of concepts, methods, technologies, and cultural changes necessary for a complete product design and manufacturing process
Weight : 0.07710117
Def: We use the term framework to refer to a structured collection of software building blocks that can be used and customized to develop components, assemble them into an application, and run the application
Weight : 0.07184533
Def: A logical structure for classifying and organizing complex information
Weight : 0.059092086
Def: A set of object classes that provide a collection of related functions for a user or piece of software
Weight : 0.055604726
Def: The software environment tailored to the needs of a specific domain
Weight : 0.046193704
Def: A component that allows its functionality to be extended by writing plug-in modules ("framework e-xtensions")
(other definition follow...)

Example 2: selecting definitory from non-definitory sentences in free texts: "**ontology alignment**" (selected sentences underlined, selected hypernym in bold)

Def: Ontology ontology alignment is not valuable for its own sake, but is worthwhile only in the service of some other function that requires it
Weight:0.03227434
Def: ontology alignment refers to the **situation**, where both the source and target ontology are known and mappings between the two ontologies are used as source for explanation
Weight:0.03170026
Def:Ontology alignment is the **automated resolution** of semantic correspondences between the representational elements of heterogenous sytems
Weight:0.026186492
Def:Ontology alignment is a foundational problem area for semantic interoperability
Weight:0.0204144
Def:ontology alignment is extreme: terms from different ontologies are always assumed to mean different things by default, and all ontology mapping is done by humans (implicitly, by putting them into the same col- umn of a report)
Weight:0.020371715
Def:Ontology alignment is also crucial for reusing the existing ontologies and for facilitating their interoperability
Weight:0.01861836
Def:Ontology alignment is also very relevant in a Semantic Web context
Weight:0.016911233
(other definition follow...)

Appendix II : An excerpt of sub-trees extracted from the INTEROP domain.

```
integration
  system_engineering_integration
  sensing_integration
  system_sensing_integration
      enterprise_system_sensing_integration
  strategy_integration
    business_strategy_integration
  software_integration
    application_integration
       enterprise_application_integration
          legacy_enterprise_
               application_integration
  service_integration
    web_service_integration
  computing_integration
    enterprise_computing_integration
  inter_organisational_integration
    enterprise_inter_organisational_integration
  intra_organisational_integration
    enterprise_intra_organisational_integration
  organization_integration
  conceptual_integration
  representation_integration
    view_integration
    ontology_integration
    model_integration
      ontology_integration
      enterprise_model_integration
      schema_integration
      scheduling_integration
          process_integration
    scheduling_integration
    design_process_integration
    business_process_integration
      on_demand_business_process_integration
    planning_integration
    enterprise_integration
  system_integration
    information_system_integration
    process_integration
      scheduling_integration
      design_process_integration
      business_process_integration

on_demand_business_process_integration
      planning_integration
      enterprise_integration
    natural_language_based_system_integration
    distributed_system_integration
```

database_system_integration
enterprise_application_integration
legacy_enterprise_application_integration
schema_integration
method_of_integration
component_integration
supply_chain_integration
human_supply_chain_integration
semantic_integration
ontology_driven_integration
information_integration
knowledge_integration
content_integration
multilingual_content_integration
enterprise_information_integration
legacy_enterprise_information_integration
intelligent_information_integration
ontology_based_integration
business_process_support_integration
database_integration
data_automatic_integration