Toward an Ecosystem of LOD in the Field: LOD Content Generation and Its Consuming Service

Takahiro Kawamura[1,2] and Akihiko Ohsuga[2]

[1] Corporate Research & Development Center, Toshiba Corp.
[2] Graduate School of Information Systems,
University of Electro-Communications, Japan

Abstract. This paper proposes to apply semantic technologies in a new domain, Field research. It is said that if "raw data" is openly available on the Web, it will be used by other people to do wonderful things. But, it would be better to show a use case together with that data, especially in the dawn of LOD. Therefore, we are proceeding with both of LOD content generation and its application for a specific domain. The application addresses an issue of information retrieval in the field, and the mechanism of LOD generation from the Web might be applied to the other domain. Firstly, we demonstrate the use of our mobile application, which searches a plant fitting the environmental conditions obtained by the smartphone's sensors. Then, we introduce our approach of the LOD generation, and present an evaluation showing its practical effectiveness.

Keywords: Sensor, LOD, AR, Plant, Field.

1 Introduction

Semantic search is intrinsically suited for information retrieval in the field, where a trial-and-error approach to search is difficult because input is less convenient and the network tends to be slower than in the case of desktop search. It is burdensome for users in the field to research something while changing keywords and looking through a list of the results repeatedly. Therefore, search with SPARQL, which can specify the necessary semantics, would be useful in the field. Moreover, exploitation of mobile and facility sensors is now prevailing, but applications are still vague although sensor information is overflowing. Thus, LOD can serve as an intermediary interpreting the semantics of the users and their environmental information obtained by the sensors and connecting it to the collective intelligence on the net. LOD and SPARQL have the tremendous potential in the field. However, to build ecosystem of LOD used in the field, it requires at least the actual LOD content for the field, and its appealing application which consumes that LOD. In the talk of Tim Berners-Lee at TED2009 and 2010 [1], it is intended that someone publishes data, and then the other one

[1] `http://www.ted.com/talks/tim_berners_lee_on_the_ next_web.html`

P. Cudré-Mauroux et al. (Eds.): ISWC 2012, Part II, LNCS 7650, pp. 98–113, 2012.

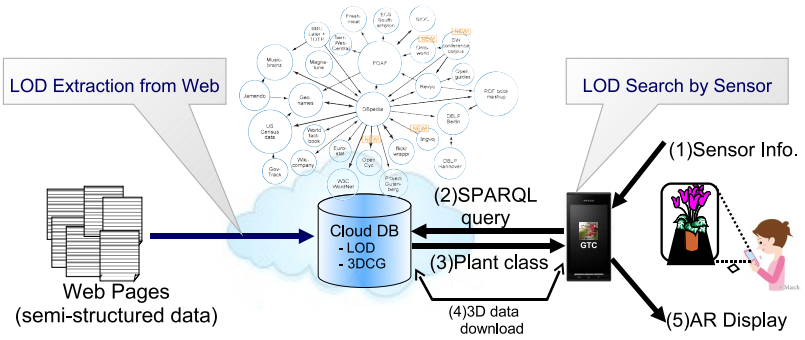

Fig. 1. Proposed architecture of LOD use

will use it, and create application mashups. But it would be better to show a use case with the data, in which the linking data is semantically utilized. Therefore, we would like to propose both of a mechanism of LOD content generation and its concrete application in this paper.

The remainder of this paper is organized as follows. Section 2 outlines related work, and then firstly we introduce a field application of LOD, where LOD is searched based on the sensor information on a smartphone in section 3. Next, section 4 describes a mechanism of LOD content generation, where LOD is extracted from the Web. Finally, section 5 presents conclusions and identifies future issues.

2 Related Work

First, we introduce two researches regarding architecture using sensors and semantics, and its application. The first one is Semantic Sensor Network (SSN), in which sensor data is annotated with semantic metadata mainly to support environmental monitoring and decision-making. SemSorGrid4Env[1] is applying it to flood emergency response planning. Our architecture (Fig. 1) is similar to SSN. However, instead of searching and reasoning within the collected semantic sensor data, we assume the existence of LOD on the net, to which the sensor data is connected. In that sence, SENSEI[2] had almost the same purpose to integrate the physical with the digital world. But the project mainly addressed the scalability issue and the definition of services interfaces, and then LOD content was limited to a few types of data like geospatial.

The second one is about social sensor research, which integrates the existing social networking services and physical-presence awareness like RFID and twitter with GPS data to encourage users' collaboration and communication. Live Social Semantics (LSS)[3] applied it to some conferences and suggested new interests for the users. It resembles our architecture in that face-to-face contact events based on RFID are connected to the social information on the net. However, from the difference in its objective, which is a social or field support, the information flow

is opposite. In our architecture, the sensor (client) side requests the LOD on the net, although in LSS the social information (DB) collects the sensor data.

Next, we introduce a research regarding LOD content generation. There are several ways and their combinations to generate LOD content. The first one is that an expert writes about a particular theme, e.g. data of Open Government. Also, there is a way to generate LOD as well as creating the content using CMS (Content Management Systems). The second and third ones are user participatory creation, e.g. DBPedia[4] and Freebase[5], and crowdsourcing, e.g. use of Amazon Mechanical Turk. Both of them use the power of the masses (or collective intelligence), but are classified according to the presence of business contract. The fourth one is conversion from the existing structured data like table, CSV and RDB using XLWrap[6] and OntoAccess[7], e.g. Life Science data, and then the last way we think is the (semi-)automatic generation of LOD from the Web. In the recent conferences, researches on the (semi-)automatic generation seems small in number, compared to LOD utilization under the premises that large-scale datasets have been provided, though there are many researches to build an ontology from the Web. But, one of them is NELL (Never-Ending Language Learner) presented by T. Mitchell at ISWC09[8] and more details at AAAI10[9], which is a semantic machine learning system using the existing ontologies, where several learning methods are combined to reduce extraction errors. Our generation method has been greatly inspired by NELL. The detailed description will be shown in section 4.

3 Field Application of LOD

3.1 Problem Statement

Home gardens and green interiors have been receiving increased attention owing to the rise of environmental consciousness and growing interest in macrobiotics. However, the cultivation of greenery in a restricted urban space is not necessarily a simple matter. In particular, as the need to select greenery to fit the space is a challenge for those without gardening expertise, overgrowth or extinction may occur. In regard to both interior and exterior greenery, it is important to achieve an aesthetic balance between the greenery and the surroundings, but it is difficult for amateurs to imagine the future form of the mature greenery. Even if the user checks images of mature greenery in gardening books, there will inevitably be a gap between the reality and the user's imagination. To solve these problems, the user may engage the services of a professional gardening advisor, but this involves cost and may not be readily available. Therefore, we considered it would be helpful if an 'agent' service offering gardening expertise were available on the user's mobile device. In this section, we describe our development of Green-Thumb Camera, which recommends a plant to fit the user's environmental conditions (sunlight, temperature, etc.) by using a smartphone's sensors. Moreover, by displaying its mature form as 3DCG using AR (augmented reality) techniques, the user can visually check if the plant matches the user's

surroundings. Thus, a user without gardening expertise is able to select a plant to fit the space and achieve aesthetic balance with the surroundings.

The AR in this paper refers to annotation of computational information to suit human perception, in particular, overlapping of 3DCG with real images. This technique's development dates back to the 1990s, but lately it has been attracting growing attention, primarily because of its suitability for recent mobile devices. AR on mobile devices realizes the fusion of reality and computational information everywhere. Research[10] on AR for mobile devices was conducted in the 1990s, but it did not attract public attention because "mobile" computers and sensors were big and hard to carry, and the network was slow.

Plant recommendation involves at least two problems. One problem concerns plant selection in accordance with several environmental conditions of the planting space. There are more than 300,000 plant species on the Earth , and around 4,000 plant species exist in Japan. Also, their growth conditions involve a number of factors such as sunlight, temperature, humidity, soil (chemical nutrition, physical structure), wind and their chronological changes. Therefore, we have incorporated the essence of precision farming[11], in which those factors are carefully observed and analyzed, and crop yields are maximized through optimized cultivation. In our research, firstly, using the sensors on the smartphone, we determine the environmental factors listed in Table 1, which we consider to be the major factors, and then try to select a plant based on those factors. Other factors, notably watering and fertilizing, are assumed to be sufficient. We intend to incorporate other factors in the near future [2].

Another problem concerns visualization of the future grown form. As well as the need to achieve aesthetic balance for both interior and exterior greenery, overgrowth is an issue. In fact, some kinds of plant cannot be easily exterminated. Typical examples of feral plants are vines such as Sicyos angulatus, which is designated as an invasive alien species in Japan, and Papaver dubium, which has a bright orange flower and is now massively propagating in Tokyo. Therefore, we propose visualization of the grown form by AR to check it in advance.

3.2 Plant Recommendation Service

This section explains the service that we propose.

Service Flow of Plant Recommendation. Firstly, the user puts an AR marker (described later) at the place where he/she wants to grow a plant , and then taps an Android application, Green-Thumb Camera (GTC), and pushes a start button. If the user looks at the marker through a camera view on the GTC App (Fig. 1), the app (1) obtains the environmental factors, such as sunlight, location and temperature from the sensor information (2) searches on LOD Cloud DB with SPARQL, and (3) receives some Plant classes that fit the environment.

[2] A bioscience researcher whom we consulted confirmed that the factors listed in Table 1 are sufficient to serve as the basis for plant recommendation to a considerable extent.

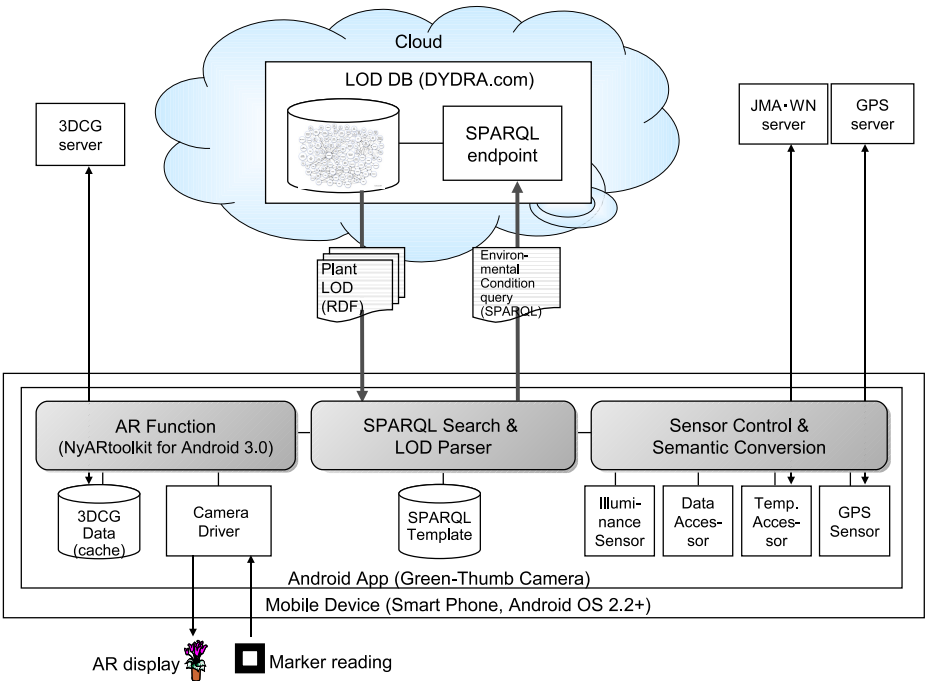

Fig. 2. Green-Thumb Camera

Then, the app (4) downloads 3DCG data for the plants, if necessary (the data once downloaded is stored in the local SD card), (5) overlays the 3DCG on the marker in the camera view. It also shows two tickers, one for the plant name and description below, and another for the retrieved sensor information on the top. Fig. 3 is an example displaying "Begonia". It is difficult to find a plant which can survive in a shade garden, so that we can find that the service is helpful. If the user does not like the displayed plant, he/she can check the next possible plant by clicking 'prev' or 'next' button, or flicking the camera view. Furthermore, if the user clicks a center button, GTC shows a grown form of the plant. In this way, the user would be able to find a plant that fits the environmental conditions and blends in with the surroundings. Fig. 2 shows the overview of this service.

Semantic Conversion from Sensor Information to Environmental Factors. This section describes the environmental factors, and how we convert raw data of the sensors to them. Table 1 shows the factors considered in this paper.

Sunlight
: This factor indicates the illuminance suitable for growing each plant and has several levels such as shade, light shade, sunny, and full sun[12,13]. To determine the current sunlight, we used a built-in illuminance sensor on the smartphone. After the application boots up, if the user brings the smart-

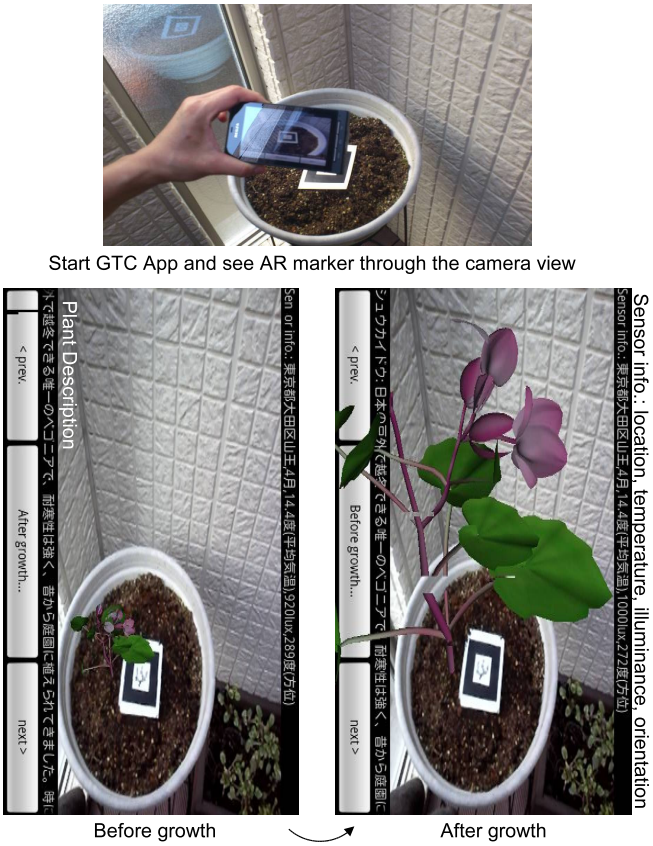

Fig. 3. Example of a plant display (left: before growth, right: after growth)

phone to the space where he/she envisages putting the plant and pushes the start button on the screen, the sunlight at the space is measured, and classified as the above levels. If it is less than 300 lux, it is deemed to be a shade area. If it is more than 300 lux but less than 3000 lux, it is deemed to be light shade, and If it is more than 3000 lux but less than 10000 lux, it is deemed to be sunny. Then, if more than 10000 lux, it is deemed to be a full sun area. In the case that the sunlight taken by the sensor fits that for the plant, it is deemed suitable.

Temperature

This factor indicates the range (min, max) of suitable temperature for a plant. The lower and the upper limits of the range are determined by reference to the sites as well as to the sunlight. To get the temperature, we referred to past monthly average temperatures for each prefecture from the Japan Meteorological Agency(JMA) [14] , using the current month and area (described below), instead of the current temperature. The temperature for

Table 1. Environmental factors

Factor	Description
Sunlight	minimum and maximum illuminance
Temperature	minimum and maximum temperature
Planting Season	optimum period of planting
Planting Area	possible area of planting

indoor plants from November to February is the average winter indoor temperature for each prefecture from WEATHERNEWS INC.(WN)[15]. In the case that the temperature taken by the sensor is within the range of the plant, it is deemed suitable.

Planting Season

The planting season means a suitable period (start, end) for starting to grow a plant (planting or sowing). The periods are set on a monthly basis according to some gardening sites[16,17]. To get the current month, we simply used the Calendar class provided by the Android OS. However, the season is affected by the geographical location (described below). Therefore, it is set one month later in the south area, and one month earlier in the north area. In the northernmost area, it is set two months earlier. , because the periods are given mainly for Tokyo (middle of Japan) on most websites. If the current month is in the planting season for the plant, it is deemed suitable.

Planting Area

The planting area means a suitable area for growing a plant. It is set by provincial area according to a reference book used by professional gardeners[12]. To get the current area, we used the GPS function on the smartphone. Then, we classified the current location (latitude, longitude) for the 47 prefectures in Japan, and determined the provincial area. If the current location is in the area for the plant, it is deemed suitable.

Recommendation Mechanism. In this section, we describe how a plant is recommended based on the above factors.

As a recommendation mechanism, we firstly tried to formulate a function on the basis of multivariate analysis, but gave it up because priority factors differ depending on the plant. Next, we created a decision tree per plant because the reasons for recommendation are relatively easily analyzed from the tree structure , and then we evaluated the recommendation accuracy[18]D However, this approach obviously poses a difficulty in terms of scaling up since manual creation of training data is costly. Therefore, we prepared Plant LOD based on collective intelligence on the net and adopted an approach of selecting a plant by querying with SPARQL.

There are several DBs of plants targeting such fields as gene analysis and medical applications. However, their diverse usages make it practically impossible to unify the schemas. Furthermore, there are lots of gardening sites for hobbyists,

and the practical experience they describe would also be useful. Therefore, instead of a Plant DB with a static schema, we adopted the approach of virtually organizing them using LOD on the cloud. Thus, we developed a semi-automatic generation system for Plant LOD, and combined the collected data with the DBPedia. The details are described in the next section.

The SPARQL query includes the above-mentioned environmental factors obtained from the sensors in the FILTER evaluation, and is set to return the top three plants in the reverse order of the planting difficulty within the types of Plant class.

It should be noted that SPARQL 1.0 does not have a conditional branching statement such as IF-THEN or CASE-WHEN in SQL. Thus, certain restrictions are difficult to express, such as whether the current month is within the planting season or not. Different conditional expressions are required for two cases such as *March* to *July* and *October* to *March*. Of course, we can express such a restriction using logical-or(||) and logical-and(&&) in FILTER evaluation, or UNION keyword in WHERE clause. But, it would be a redundant expression in some cases (see below, where ?start, ?end, and MNT mean the start month, the end month, and the current month respectively). On the other hand, SPARQL 1.1 draft[19] includes IF as Functional Forms. So we expect the early fix of 1.1 specification and dissemination of its implementation.

```
SELECT distinct ...
WHERE{
...
FILTER(
...
&&
# Planting Season
( ( (xsd:integer(?start) <= MNT) && (MNT <= xsd:integer(?end)) ) ||
  ( (xsd:integer(?start) >= xsd:integer(?end)) &&
    (xsd:integer(?start) <= MNT) && (MNT <= 12) ) ||
  ( (xsd:integer(?start) >= xsd:integer(?end)) &&
    ( 1 <= MNT) && (MNT <= xsd:integer(?end)) ) )
&&
..
)
ORDER BY ASC (xsd:integer(?difficulty))
LIMIT 3
```

Listing 1.1. SPARQL query

AR Interface. This application requires a smartphone running Google Android OS 2.2+ and equipped with a camera, GPS, and a built-in illuminance sensor. For the AR function, we used NyARToolkit for Android[20], which is an AR library for the Android OS using a marker. It firstly detects the predefined marker (Fig. 4) in the camera view, recognizes its three-dimensional position and attitude, and then displays 3DCGs in Metasequoia format on the marker. The 3DCG can quickly change its size and tilt according to the marker's position and attitude through the camera. We have already prepared 90 kinds of plant 3DCG data for recommendation.

Fig. 4. AR marker (6cm × 6cm)

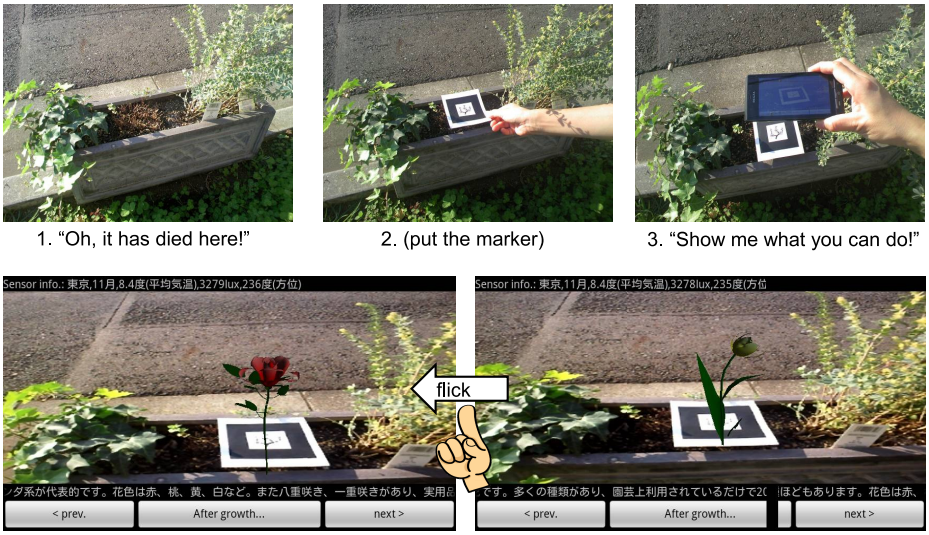

Fig. 5. Result of plant recommendation

3.3 Experimental Result of LOD Application

Fig. 5 shows an experimental result of the plant recommendation. The test environment was as follows: Tokyo, November, 3000+ lux, approx. 10 °C. If the user puts the marker at a place where he/she envisages putting a plant, and sees it through the camera, the GTC App reads the marker and gets the environmental factors such as sunlight, location, and temperature. Then, it overlays 3DCG of a recommended plant on the marker in the camera view. If the user views the marker from different angles and distances through the camera, it dynamically changes the 3DCG as if it were the real thing. Also, by flicking the camera view, the next plant in the order of recommendation is displayed.

In the figure, 3DCG of a rose and a tulip are displayed as a result. Those are typical candidates for planting in this season in Tokyo, and we confirmed the recommendation is working correctly. The GTC App is now open to the public, so anyone can download and try to use it [3]. In the near future, we are

[3] `http://www.ohsuga.is.uec.ac.jp/~kawamura/gtc.html` (in Japanese).

planning to conduct some evaluation by a group of potential users to determine it's effectiveness.

4 LOD Content Generation

4.1 Overview of Plant LOD

The Plant LOD (Fig. 6) is RDF data, in which each plant is an instance of "Plant" class of DBpedia ontology. DBpedia has already defined 10,000+ plants as types of the Plant class and its subclasses such as "FloweringPlant", "Moss" and "Fern". In addition, we created 90 plants mainly for species native to Japan. Each plant of the Plant class has almost 300 Properties, but most of them are inherited from "Thing", "Species" and "Eukaryote". So we added 19 properties to represent necessary attributes for plant cultivation, some of which correspond to { Japanese name, English name, country of origin, description, sunlight, temperature (min), temperature (max), planting season (start), planting season (end), blooming season (start), blooming season (end), watering amount, annual grass (true or false), related website, image URL, 3DCG URL, planting area, planting difficulty }. { name, country of origin, description, sunlight, temperature, planting season, blooming season, watering amount, planting difficulty }. Fig. 6 illustrates the overall architecture of the Plant LOD , where prefixes **gtc:** and **gtcprop:** mean newly created instances and properties. The Plant LOD is now stored in a cloud DB, DYDRA[4] and a SPARQL endpoint is offered to the public.

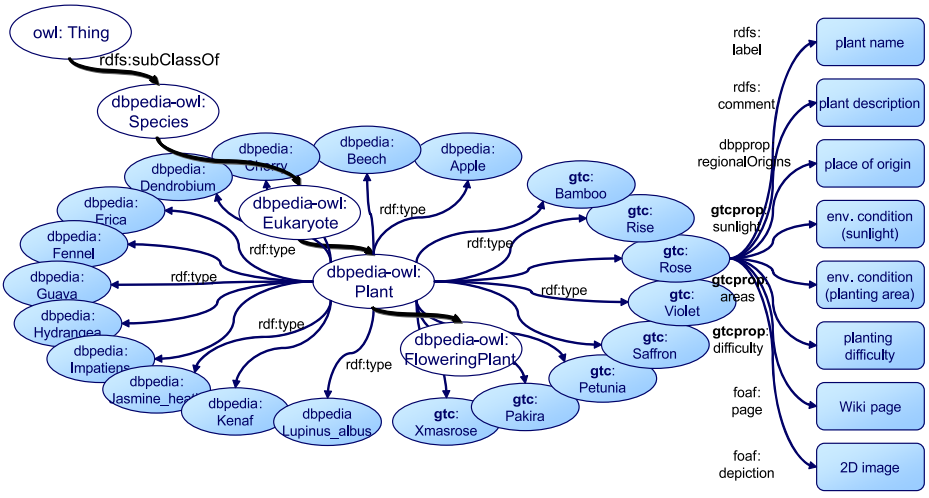

Fig. 6. Overview of Plant LOD

[4] http://dydra.com/

4.2 Semi-automatic Generation of LOD

In order to collect the necessary plant information from the Web and correlate it to the DBPedia, we developed a semi-automatic mechanism to grow the existing LOD, which involves a boot strapping method[21] based on ONTOMO[22] and a dependency parsing based on WOM Scouter[23]. But the plant names can be easily collected from a list on any gardening site and we have already defined the necessary properties based on our service requirements. Therefore, what we would like to collect in this case is the value of the property for each plant.

Process of our LOD generation is as follows (Fig. 7). First of all, we make a keyword list, which includes an instance name (plant name) and a logical disjunction of property names such as Rosemary ("Japanese name" OR "English name" OR "country of origin" OR ...), and then search on Google , and receive the result list, which includes more than 100 web pages. Next, we crawl the pages except for pdf files and also check Google PageRank for each page.

As the boot strapping method, we first extract specific patterns of DOM tree from a web page based on some keys, which are the property names (and their synonyms), and then we apply that patterns to other web pages to extract the values of the other properties. This method is mainly used for extraction of $(property, value)$ pairs from structured part of a page such as tables and lists (Fig. 8 left).

However, we found there are many (amateur) gardening sites that explain the nature of the plant only in plain text. Therefore, we developed an extraction method using the dependency parsing, because a triple $< plantname, property, value >$ corresponds to $< subject, verb, object >$ in some cases. It first follows modification relations in a sentence from a seed term, which is the plant name or the property name (and their synonyms), and then extract the triple, or a triple $< -, property, value >$ in the case of no subject in the sentence ('-' is replaced with the plant name in the keyword list later). See Fig. 8 right.

We combine all the property values obtained above, and filter it if it matches to co-occurrence strings with the corresponding property names, where a set of the co-occurrence string are prepared in advance, e.g. the propery "temperature" obviously co-occurs with a string °C. Then, we form some clusters of the identical property values for each property name based on LCS (Longest Common Substring). Furthermore, for correction of errors, which may be an error of extraction and/or of the information source, we sum up the PageRanks of the source pages for each cluster to determine the best possible property value and the second-best. Finally, after a user determines a correct value from the proposed ones, CSV and RDF files are generated to each plant.

In either way, the key or seed of the boot strapping and the dependency parsing are retrieved from our predefined schema of Plant LOD, that is, the instance name and the property name. It is our idea to put flesh on the bones of the existing LOD like the DBPedia in order to correlate to it.

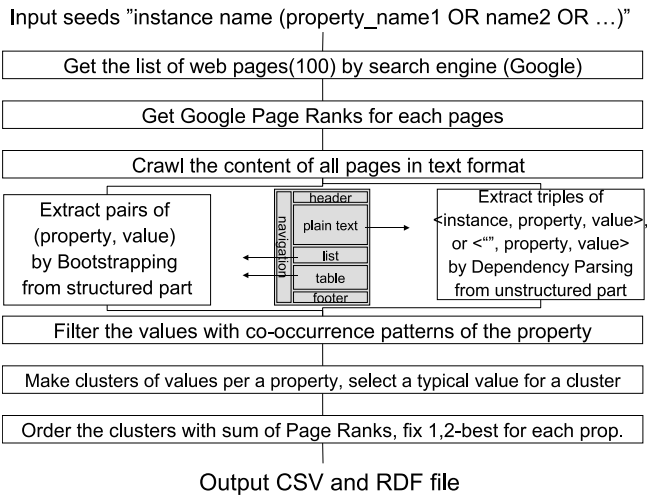

Fig. 7. Process of LOD content generation

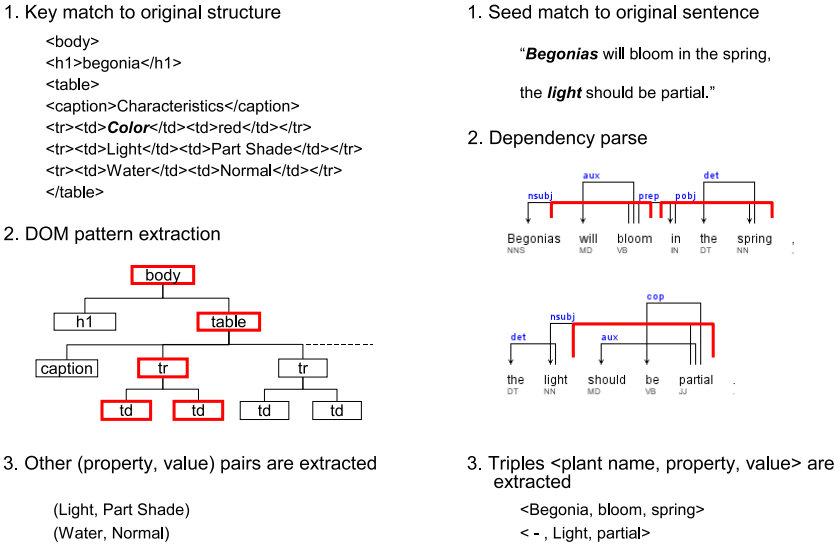

Fig. 8. Examples of boot strapping and dependency parsing

4.3 Evaluation of LOD Generation

Extraction Accuracy. We applied this LOD generation mechanism to extract the values of the 13 properties for the 90 plants that we added. The result shown in Table 2 includes an average presicion and recall of the best possible value (1-best) obtained through the whole process, the boot strapping method only, and

Table 2. Extraction accuracy

Accuracy(%)	1-best	2-best	1-best (bootstrap only)	1-best (dependency only)
Precision	85.2	97.4	88.6	85.2
Recall	76.9	87.2	46.2	76.9
Amount Ratio (%)	–	–	10.8	89.2

the dependency parsing only, and then these of the second possible value (2-best). It should be noted that we collected 100 web pages for each plant, but some reasons such as DOM parse errors and difference of file types reduced the amount to about 60%. In terms of determining the seasons (start month and end month), if the extracted period is subsumed by the correct period, and the gap between the start (end) months is within 1 month, then it is regarded as correct. Also, in terms of the temperature, if the gap is within 3 °C, it is regarded as correct. Properties like description, which are not clear whether it is true or not, are out of scope of this evaluation. If there are more than two clusters whose sum of the PageRanks are the same, we regarded them all as the first position. The accuracy is calculated in units of the cluster instead of each extracted value. That is, in the case of 1-best, a cluster which has the biggest PageRank corresponds to an answer for the property. In the case of 2-best, two clusters are compared with the correct value, and if either one of the two answers is correct, then it is regarded as correct (thus, it is slightly different than average precision).

$$N - best\ precision = \frac{1}{|D_q|} \sum_{1 \leq k \leq N} r_k$$

,where $|Dq|$ is is the number of correct answers for question q, and r_k is an indicator function equaling 1 if the item at rank k is correct, zero otherwise. The boot strapping method only and the dependency parsing only mean to form the clusters out of the values extracted only by the boot strapping and the dependency parsing, respectively. A cluster consists of the extracted values for a property, which seem identical accroding to LCS, but the number of the values in a cluster may vary from more than 10 to 1. Finally, if there are various theories as to the correct value for a property, we selected the most dominant one.

The best possible values (1-best) achieved an average precision of 85% and an average recall of 77%. But, the 2-best achieved an average precision of 97% and an average recall of 87%. So if we are permitted to show a binary choice to the user, it would be possible to present a correct answer in them in many cases. The accuracy of the automatic generation would not be 100% after all, and then a human checking is necessary at any step. Therefore, the binary choice would be a realistic option.

In detail, the boot strapping collects smaller amounts of values (11%), so the recall is substantially lower (46%) than the dependency parsing, but the precision is higher (89%). It is because data written in the tables can be correctly extracted, but lacks diversity of properties. Semantic drift of the values extracted by generic

patterns, which is a well-known problem in the boot strapping method, rarely happened here, because target sources are at most top 100 pages of the Google result, and the values are sorted by the PageRank at the end.

On the other hand, the dependency parsing collects large amount of values (89%), but it is a mixture of wheat and chaff. But, the total accuracy is affected by the dependency parsing, because the biggest cluster of the PageRank is composed mainly of the values extracted by the dependency parsing. So we will consider to put some weight on the values extracted by the boot strapping.

In addition, the accuracy varies by kind of the plant and the properties. The popularity of a plant affects the quantity and quality of information sources (web pages). Also, a specific property like "temperature (max)" is a minor information and the absolute number of descriptions are small.

Extraction Approach. In comparison with the above-mentioned NELL, our mechanism is the same as the NELL, as far as it is "Ontology-driven", "Macro-reading" which means that the input is a large text collection and the desired output is a large collection of facts, using "Machine learning methods". Recently, there have been several researches to extract information from semi-structured textual documents on the Web, which are combining NLP (Natural Language Processing) mechanisms or tools, and then using semantic resources like fine-grained ontologies. Among them, NERD (Named Entity Recognition and Disambiguation) framework[24] has also proposed an RDF/OWL-based NLP Interchange Format (NIF) and an API to unify various tools for a qualitative comparison. Both of the NELL and our mechanism are those of such researches.

However, instead of CPL (Coupled Pattern Learner) in NELL, we used a morphological analysis and a dependency parsing. Moreover, clustering of the values using LCS and the PageRank are also our own methods. But, a key strategic difference is a target domain of LOD generation. The NELL is targeting the world, so the granularity of the properties is big and the number of them is limited. For example, "agricultural product growing in state or province" is a barely fine-grain property in the NELL, but only 10 instances have that property, and also the number of all the properties is 5% of the total extraction. On the other hand, by restricting the domain of interest, the plant in this case, it is possible for our mechanism to construct the set of the co-occurrence string with the predefined property name. This simple heuristics effectively filter out candidates for the property values, thus raise the accuracy of the extraction and keep the variety of the properties together with the above methods. It obviously restricts applicability of the proposed technique, but is practically effective to generate useful LOD content.

Furthermore, the purpose of our mechanism is to grow the existing LOD in a specific domain, so that we extracted the correlating values according to the predefined schema from the Web. NELL also has the schema for broader domain than ours. In contrast, LODifier[25] has recently proposed a translation of textual information in its entirety into a structural RDF in open-domain scenarios with no predefined schema. In the future, we could use that technique as a pre-process of the whole document before matching the schema.

5 Conclusion and Future Work

Under the vision that LOD is suited for information retrieval in the field, we propose a mechanism of LOD content generation for a domain and its application to indicate the possible benefit of field use.

In the near future, we would like to apply this architecture of environmental sensing → semantic conversion → LOD Cloud (← collective intelligence) to more serious problems that would benefit from greater IT support. Now we are considering the provision of support for greening business, which addresses environmental concerns, and for agri-business in regard to the growing food problem. Also, we are planning to apply this architecture in the other fields than the plant. For example, by using a built-in GPS, and acceleration and orientation sensor of the smartphone, it is possible to understand users' outdoor behaviors, especially situation of their movements like walk, bus and train. Therefore, if traffic situation and accident data are provided as LOGD (Linking Open Government Data), we can mashup an application which shows safe navigation and precautions for the elderly people moving outside.

References

1. Gray, A.J.G., García-Castro, R., Kyzirakos, K., Karpathiotakis, M., Calbimonte, J.-P., Page, K., Sadler, J., Frazer, A., Galpin, I., Fernandes, A.A.A., Paton, N.W., Corcho, O., Koubarakis, M., De Roure, D., Martinez, K., Gómez-Pérez, A.: A Semantically Enabled Service Architecture for Mashups over Streaming and Stored Data. In: Antoniou, G., Grobelnik, M., Simperl, E., Parsia, B., Plexousakis, D., De Leenheer, P., Pan, J. (eds.) ESWC 2011, Part II. LNCS, vol. 6644, pp. 300–314. Springer, Heidelberg (2011)
2. Presser, M., Barnaghi, P.M., Eurich, M., Villalonga, C.: The SENSEI project. IEEE Communications Magazine 47(4) (2009)
3. Szomszor, M., Cattuto, C., Van den Broeck, W., Barrat, A., Alani, H.: Semantics, Sensors, and the Social Web: The Live Social Semantics Experiments. In: Aroyo, L., Antoniou, G., Hyvönen, E., ten Teije, A., Stuckenschmidt, H., Cabral, L., Tudorache, T. (eds.) ESWC 2010, Part II. LNCS, vol. 6089, pp. 196–210. Springer, Heidelberg (2010)
4. Auer, S., Bizer, C., Kobilarov, G., Lehmann, J., Cyganiak, R., Ives, Z.G.: DBpedia: A Nucleus for a Web of Open Data. In: Aberer, K., Choi, K.-S., Noy, N., Allemang, D., Lee, K.-I., Nixon, L.J.B., Golbeck, J., Mika, P., Maynard, D., Mizoguchi, R., Schreiber, G., Cudré-Mauroux, P. (eds.) ASWC 2007 and ISWC 2007. LNCS, vol. 4825, pp. 722–735. Springer, Heidelberg (2007)
5. Freebase, `http://www.freebase.com/`.
6. Langegger, A., Wöß, W.: XLWrap – Querying and Integrating Arbitrary Spreadsheets with SPARQL. In: Bernstein, A., Karger, D.R., Heath, T., Feigenbaum, L., Maynard, D., Motta, E., Thirunarayan, K. (eds.) ISWC 2009. LNCS, vol. 5823, pp. 359–374. Springer, Heidelberg (2009)
7. Hert, M., Ghezzi, G., Würsch, M., Gall, H.C.: How to "Make a Bridge to the New Town" Using OntoAccess. In: Aroyo, L., Welty, C., Alani, H., Taylor, J., Bernstein, A., Kagal, L., Noy, N., Blomqvist, E. (eds.) ISWC 2011, Part II. LNCS, vol. 7032, pp. 112–127. Springer, Heidelberg (2011)

8. Mitchell, T.M., Betteridge, J., Carlson, A., Hruschka, E., Wang, R.: Populating the Semantic Web by Macro-reading Internet Text. In: Bernstein, A., Karger, D.R., Heath, T., Feigenbaum, L., Maynard, D., Motta, E., Thirunarayan, K. (eds.) ISWC 2009. LNCS, vol. 5823, pp. 998–1002. Springer, Heidelberg (2009)
9. Carlson, A., Betteridge, J., Kisiel, B., Settles, B., Hruschka Jr., E.R., Mitchell, T.M.: Toward an Architecture for Never-Ending Language Learning. In: Proc. of Conference on Artificial Intelligence, AAAI (2010)
10. Nagao, K.: Agent Augmented Reality: Agents Integrate the Real World with Cyberspace. In: Ishida, C.T. (ed.) Community Computing: Collaboration over Global Information Networks (1998)
11. Srinivasan, A.: Handbook of precision agriculture: principles and applications. Routledge (2006)
12. NEO-GREEN SPACE DESIGN. Seibundo Shinkosha Publishing (1996)
13. engeinavi (in Japanese), http://www.engeinavi.jp/db/
14. Japan Meteorological Agency, http://www.jma.go.jp/jma/indexe.html
15. weathernews, http://weathernews.com/?language=en
16. MAKIMO PLANT (in Japanese), http://www.makimo-plant.com/modules/maintenance/index.php
17. Angyo: The Village of Garden Plants, http://www.jurian.or.jp/en/index.html
18. Kawamura, T., Mishiro, N., Ohsuga, A.: Green-Thumb Phone: Development of AR-based Plant Recommendation Service on Smart Phone. In: Proc. of International Conference on Advanced Computing and Applications, ACOMP (2011)
19. W3C: SPARQL 1.1 Query Language Working Draft, May 12 (2011), http://www.w3.org/TR/2011/WD-sparql11-query-20110512/
20. NyARToolkit for Android, http://sourceforge.jp/projects/nyartoolkit-and/
21. Brin, S.: Extracting Patterns and Relations from the World Wide Web. In: Atzeni, P., Mendelzon, A.O., Mecca, G. (eds.) WebDB 1998. LNCS, vol. 1590, pp. 172–183. Springer, Heidelberg (1999)
22. Kawamura, T., Shin, I., Nakagawa, H., Nakayama, K., Tahara, Y., Ohsuga, A.: ONTOMO: Web Service for Ontology Building - Evaluation of Ontology Recommendation using Named Entity Extraction. In: Proc. of IADIS International Conference WWW/INTERNET 2010, ICWI (2010)
23. Kawamura, T., Nagano, S., Inaba, M., Mizoguchi, Y.: Mobile Service for Reputation Extraction from Weblogs - Public Experiment and Evaluation. In: Proc. of Twenty-Second Conference on Artificial Intelligence, AAAI (2007)
24. Rizzo, G., Troncy, R., Hellmann, S., Bruemmer, M.: NERD meets NIF: Lifting NLP Extraction Results to the Linked Data Cloud. In: Proc. of 5th Workshop on Linked Data on the Web, LDOW (2012)
25. Augenstein, I., Padó, S., Rudolph, S.: LODifier: Generating linked data from unstructured text. In: Simperl, E., Cimiano, P., Polleres, A., Corcho, O., Presutti, V. (eds.) ESWC 2012. LNCS, vol. 7295, pp. 210–224. Springer, Heidelberg (2012)