[White Paper]

T-Engine Forum
Ubiquitous ID Center
Document
DRAFT

910-S002-0.00.24 / UID-CO00002-0.00.24
11/16/2006

Ubiquitous ID Architecture

Number: 910-S002-0.00.24 / UID-CO00002-0.00.24
Title: Ubiquitous ID Architecture
Status: [X] Working Draft, [ ] Final Draft for Voting, [ ] Standard
Date: 11/16/2006

Table of Contents

Ubiquitous ID Architecture

Introduction

Scope

This document describes the basic concept of Ubiquitous ID architecture. It also specifies an outline of the fundamental technologies for Ubiquitous ID architecture.

Position of this Document

This document describes the basic concept and system outline of Ubiquitous ID architecture. It lays the groundwork for other specifications for this ubiquitous ID architecture. Other respective documents for Ubiquitous ID architecture are derived from this specification.

References

[1] T-Engine Forum, Ubiquitous ID Center, "ucode: Ubiquitous Code." 910-S101/UID-00010, 2006.

[2] T-Engine Forum, Ubiquitous ID Center, "ucode Tag System." 930-S201/UID-00017, 2006.

[3] T-Engine Forum, Ubiquitous ID Center, "ucode Resolution Protocol (Simplified Edition)." 910-S202/UID-0005, 2006.

[4] T-Engine Forum, Ubiquitous ID Center, "ucode Resolution Protocol (Standard Edition)." 930-S221/UID-00008, 2006.

[5] T-Engine Forum, Ubiquitous ID Center, "ucode Contents Transfer Protocol: uCTP." 930-S301/UID-00009, 2006.

[6] T-Engine Forum, Ubiquitous ID Center, "ucR format: ucode Relation Description Format." 940-S101/UID-00026, 2006.

[7] T-Engine Forum, Ubiquitous ID Center, "ucR vocabulary: Vocabulary Definition in the ucR Model." 940-S301/UID-00029, 2006.

Definition of Terms

Relevant terms are defined in the text when they appear for the first time.

1. Ubiquitous ID Architecture

This chapter describes the overall outline specifications of Ubiquitous ID architecture and the structure of this document.

1.1. Purpose of Ubiquitous ID Architecture Establishment

Ubiquitous computing is a technology that supports a better way of living by providing computing cpacity to various entities in the living space—for example, walls, furniture, floors as well as home electronic appliances—and equipping them with autonomous control so that they can exchange information and operate in collaboration (Figure 1).

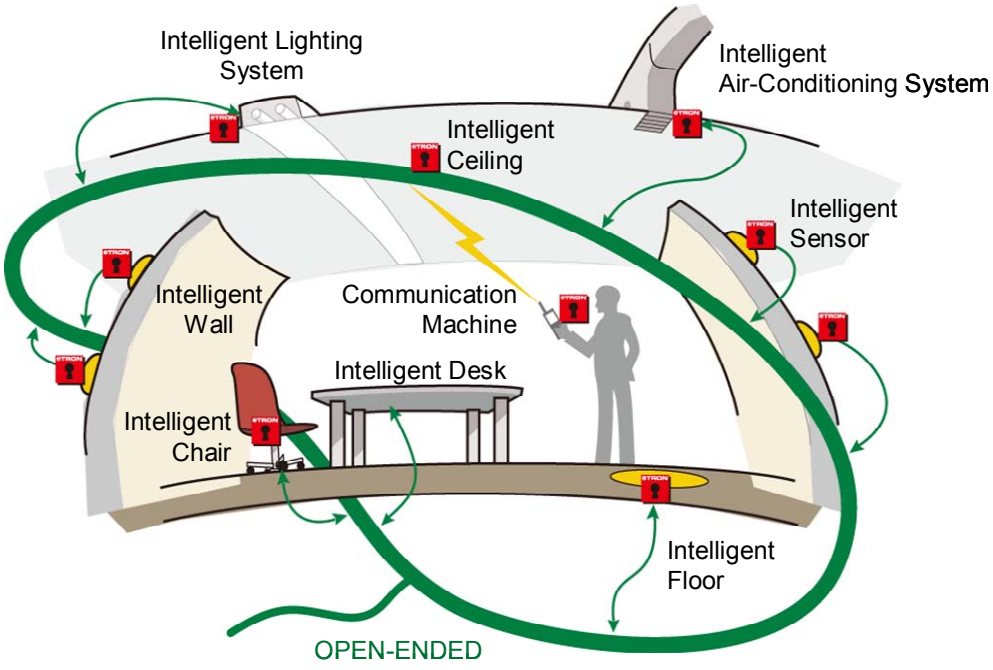

Figure 1: Image of Indoor Ubiquitous Computing

Moreover, this technology supports a better way of living by providing computing power to not only indoor but also outdoor objects including utility poles and signboards in towns and equipping them with autonomous control so that they can exchange information and operate in collaboration.

Figure 2: Image of Outdoor Ubiquitous Computing Environment

In order to realize such ubiquitous computing environment, it is important to recognize the real-world **context**. This is called **context-awareness**. To realize context-awareness, it is essential to recognize various objects, places, and concepts in the real world. It is impossible to enumerate all the contexts in the real world in advance.

Accordingly, in our approach, unique identifiers consisting of fixed-length integers are provided to respective objects, spaces, and concepts desired to be identified in the real world. We also provide frameworks that can map the real world and its context to the format acceptable as digital information by representing them as the relationships among objects, places, and concepts to which their unique identifiers are assigned. The target to be identified is called an "**entity**" in our terminology, and the unique identifier to identify the entity is called a **ucode (ubiquitous code**). To summarize, the **ucode model** is one of descriptive models representing the real-world context as digital information. This model assigns ucodes to individual entities and maps the real world and its context to the format acceptable as digital information by using the relationships between ucodes. **Ubiquitous ID architecture** is a system architecture for substantiating the ucode model.

Ubiquitous ID architecture is a common platform which obtains an device status and controls the device and provides information and services by using ucodes that identify entities in the real world as triggers. It is the infrastructure architecture to connect the real world which entities represent and the virtual world which device networks and information services represent. This document and its derived specifications are designed to realize Ubiquitous ID architecture.

1.2. Basic Principles of Ubiquitous ID Architecture

The basic principles of Ubiquitous ID architecture are as follows:

(1) Identifying an Entity

The ucode is defined as an identifier system for identifying individual entities uniquely.

(2) Assuming Network Environment

Only ucodes are assigned to individual entities themselves. Information of entities to which the ucodes are assigned is usually stored into the server on the network. By detaching entity identifications from information management, we can realize such services as obtaining the latest information on a certain object or its related entities.

(3) Providing a Security Mechanism

A secure wide-area distributed system that can assure privacy protection into account is established by using the eTRON architecture as security infrastructure.

(4) Providing an Open Platform

The Ubiquitous ID architecture specification shall be basically released.

1.3. Fundamental Technology and Mechanism of Ubiquitous ID Architecture

The fundamental technologies for realizing Ubiquitous ID architecture are as follows:

(1) ucode Tag

A ucode tag is a medium that stores a ucode.

(2) Ubiquitous Communicator: UC

A UC is a user terminal that reads ucodes and provides services based on the ucode to a user.

(3) ucode Relation Database

A ucode relation database manages entity information in a distributed manner.

(4) ucode Information Server

A ucode information server manages information displayed by and services provided to UCs.

With these fundamental technologies, a UC receives services based on the ucode in the following steps.

(1) A UC reads the ucode assigned to an entity and obtains the context in the surrounding real world.

(2) A UC accesses the ucode relation database and derives the information according to the ucode and context from the ucode relation database based on the obtained ucode. This process is called **ucode resolution**. A UC can register the information associated with ucode to the ucode relation database. This process is called **ucode registration**.

(3) As a result of the ucode resolution, a UC accesses a ucode information server after it obtains the address of the ucode information server from the ucode relation database. This allows the terminal to provide the services related to the ucode retrieved from the ucode information server such as displaying the information on the obtained ucode, and controlling the device corresponding to the ucode. Registration to the ucode relation database is among the services related to ucodes.

Figure 3 illustrates one of the important Ubiquitous ID architecture applications, which is a mechanism of deriving and displaying the information related to the obtained ucode.

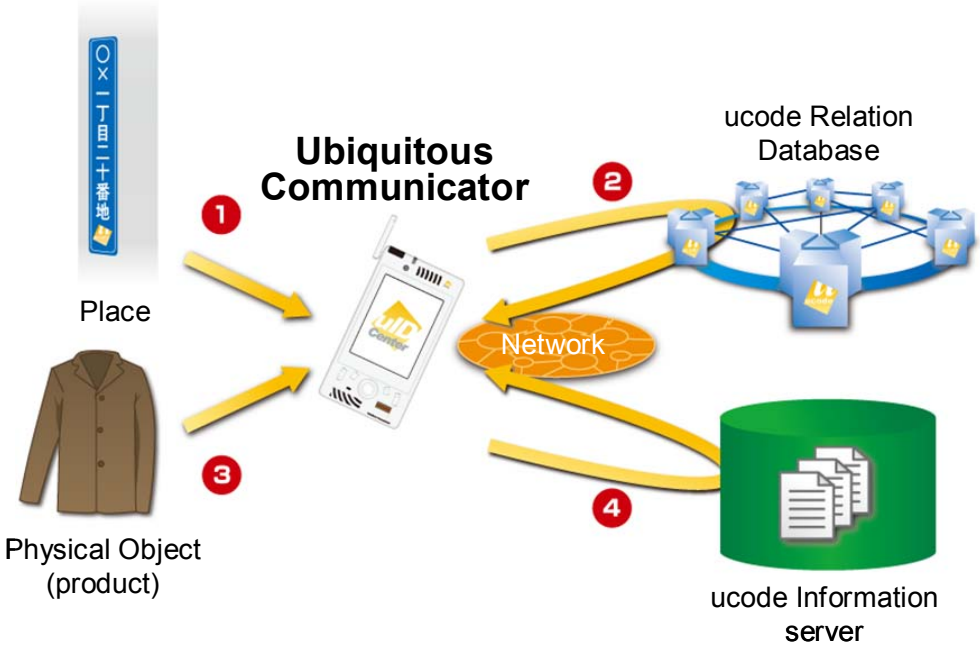

Figure 3: Information Retrieval Mechanism Based on Ubiquitous ID Architecture

2. ucode

This chapter specifies the ucode which is the base of Ubiquitous ID architecture. Among the ucode specifications are the ucode model which maps the real-world context to the format acceptable as digital information using ucodes, the ucode classification and code system, and the ucode tag which is a medium that stores a ucode. This document outlines the ucode classification and code system and the ucode tag which is a medium that stores a ucode.

2.1. ucode Model

To identify individual objects, spaces, and concepts in the real world, unique identifiers are assigned to respective objects, spaces and concepts that we wish to identify. This model is called a **ucode model**. The target in the real world to be identified in the ucode model is called an "**entity**," and the unique identifier assigned for identifying the entity is called a **ucode**.

The entity consists of objects, places, and concepts. "Objects" in the ucode model include tangible objects such as industrial products and agricultural crops, and intangible objects such as content and programs. "Places" include features in the real world such as roads and buildings and more detailed smaller components in the real world such as rooms and corridors. "Concepts" include the relationships between "objects" and "places" and the information which can be the real-world context. In a nutshell, the entity is a target to be identified among objects, places, or concepts in the real world, and the identifier to identify the entity is a ucode.

ucodes are not assigned to every entity in the real world in advance. When realizing various services, ucodes are assigned as needed to the targets needed to be identified. Conversely, objects, places, and concepts in the real world cannot be distinguished based in the ucode model unless ucodes are assigned to them. The information to which ucodes are not assigned and constant is called an **atom**. Figure 4 illustrates the above relationships.[2]

[2] It is our policy not to attach ucode tags to people due to security and private concerns.

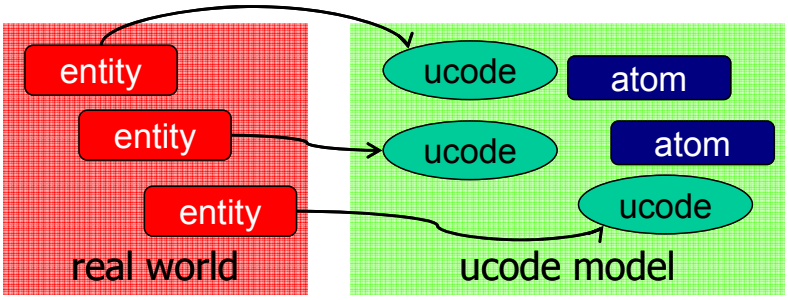

Figure 4: Relationships among ucodes, Entities and Atoms

2.2. ucode Classification

A ucode model classifies a ucode into the following two types (Figure 5).

- **Physical ucode:** A **physical ucode** is a ucode stored in the device physically associated with an individual item (this device is called a **ucode tag** [2]).
- **Logical ucode:** A **logical ucode** is not a physical ucode, in other words, does not have to be stored in a ucode tag.

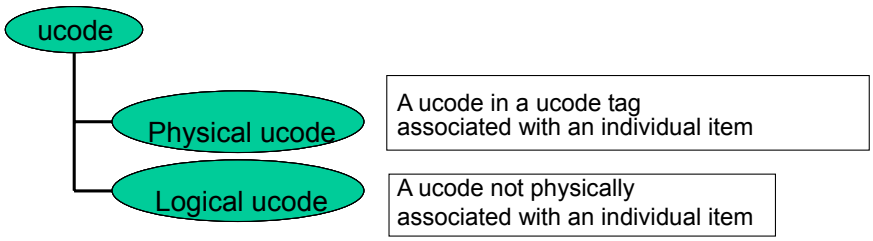

Figure 5: ucode Classification

Generally, the ucodes which identify tangible objects are physical ucodes, and the ucodes which identify intangible objects, contexts, and relationships between ucodes are logical ucodes.

2.3. Code System of ucode

The basic width of a ucode is 128 bits, and it can be extended in 128 bit units. This ucode space is managed by being divided into subspaces called a domain. In other words, a domain is a subspace as a ucode management unit . A domain can be a metacode domain that embeds other code systems. The detailed code system of a ucode is specified in [1].

2.4. ucode Tag

The medium which stores a physical ucode is called a **ucode tag**. In other words, the medium stores a ucode and is physically associated with an individual item. A detailed ucode tag system is specified in [2]. This document describes its outline.

2.4.1. ucode Tag System

Since the ucode tag is attached to a tangible object, it is influenced by the physical or RF property of the object or environment where the ucode tag is attached and used. For example, certain RFID tags cannot perform communication if they are attached to a moisture-containing material or to metallic surfaces. Moreover, depending on the application usage, it may be necessary to keep the ucode and additional information stored in the ucode tag confidential.

Therefore, Ubiquitous ID architecture does not uniquely specify the ucode tag to be used. More specifically we adopt a policy that an appropriate tag can be selected from various types of tags according to the viewpoint of merits and demerits and application requirements. In other words, the classification criteria for the ucode tag are established. The tag manufacturers then apply for certification of tags, and the tag which meets the criteria is certified as a standard ucode tag. There are two classification criteria. One is a security class, and the other is an interface category.

2.4.2. Security Class

The security function is classified according to the security and privacy protection function which tags must be equipped with and consists of the following seven classes.

- **Class 0:** Data defect detection function
  Damage caused in a part of data due to disturbance or physical defects in tags can be detected.
- **Class 1:** Physical duplication resistance and physical forgery resistance function
  Creating data which is physically identical or similar is difficult.

- **Class 2:** Identification prevention function
  Unauthorized third parties are prevented from identifying ucodes stored in tags and communication content, status, and methods.
- **Class 3:** Tamper resistance, access control management function by resources Information stored in tags cannot be illegally read physically or logically.
- **Class 4:** Secure communications function with unknown nodes
  A secure data communication path can be established even for an unidentified node that does not share a private key in advance via an open network.
- **Class 5:** Time-dependent resource management function
  Time-limit management for carrier data, security information, and tag operation can be conducted, such as setting up a data validity period and stopping operations after a certain period of time.
- **Class 6:** Internal programs and security information update function
  Protection function permits maintaining an optimum security status based on their usage, such as updating firmware and applying security patches.

2.4.3. Interface Category

The interface is classified according to a tag interface device equipped in a UC and consists of the following four categories.

- **Category 0:** Print tags (bar codes, two-dimensional bar codes, etc.)
- **Category 1:** Passive RF tags (RFID tags with contactless interfaces, contactless IC cards, etc.)
- **Category 2:** Active RF tags
- **Category 3:** Active infrared tags

Ubiquitous ID architecture certifies and tags that satisfy the following conditions as a standard ucode tag, and use them.

(1) Numbering and operations which assure the uniqueness of ucodes are provided.
(2) Any of the security classes from Class 0 to 6 is satisfied.
(3) Technical information for communicating with UCs is disclosed.

3. Information Representation by ucode

This chapter outlines information representation by using ucodes. The specifications for information representation by using ucodes are as follows:

- the ucode Relation model (ucR model) which represents information by using the relationships between ucodes,
- the ucR format which is a representation specification for the ucR model, and
- the ucR vocabulary which defines the allocation of the meaning to the basic logical ucodes that should be understood by each application in a common way.

This document outlines the specifications of ucR model, ucR format, and ucR vocabulary. Details of the ucR format are defined in [6] and ucR vocabulary in [7] respectively.

3.1. ucode Relation Model (ucR model)

Ubiquitous ID architecture represents the real-world context by modeling the relation on a real-world entity as a relational representation among ucodes allocated to entities or between the ucode and atom. This representation model is called a **ucode Relation model** (ucR model). When a relationship is represented by ucR model, it is called a RELATION.

3.1.1. ucR Unit

The basic unit of the ucR model consists of two ucodes or a ucode and atom, and their relationships. Moreover, the respective relationships between the two are provided with a logical ucode called a relation ucode. This basic unit consisting of the above triplet is called a ucode relation unit (**ucR unit**). When this triplet is set in a sentence where the relationships are represented as the predicate, the ucode as the subject is called a **subject ucode**, and the ucode as the predicate which is a relation part is called a **relation ucode**, and the ucode as the object or complement is called an **object ucode**. The atom can be substituted for the object ucode. For example, in a description, "R of S is O" or "there is a relation R between S and O," S refers to the subject ucode, R to the relation ucode, and O to the object ucode or atom. When it is not necessary to uniquely identify any of the three elements comprising a ucR unit,

their relevant parts can be left blank.

A ucR unit can be drawn as diagram in the following manner (Figure 6).

- ucodes are represented as the shape of ellipses, and ucode values or alias names are entered in the ellipses.
- Atoms are represented as rectangles, and their details are entered in the rectangles.
- Blank elements are represented as the shape of ellipses, and the insides of the ellipses are left blank.
- Arrows connect subject ucodes and an object ucode or an atom in the direction from the subject ucode to the object ucode or the atom, and relation ucodes are placed on the arrows.

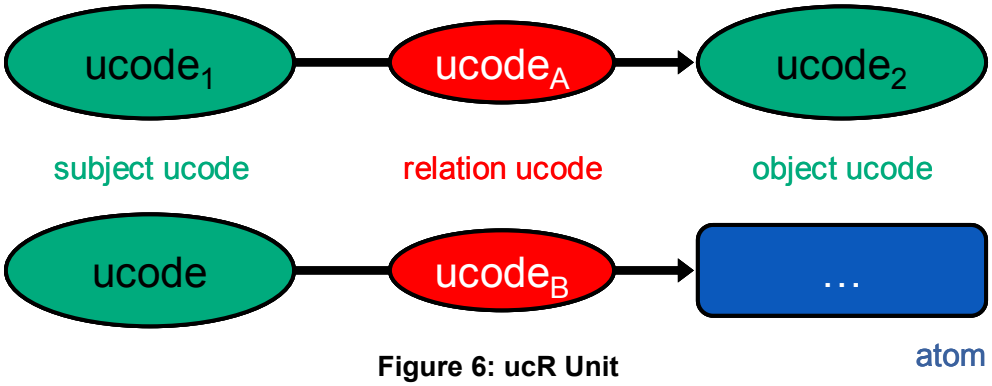

Figure 6: ucR Unit

The real-world context which is used in ubiquitous computing can be represented by using these ucR units. The real-world context is represented in Ubiquitous ID architecture by using this ucR units.

A simple representation example of the real world context by using ucR units is shown below. Figure 7 illustrates a diagram of each ucR unit drawn using the procedures above.

(1) Suppose ucode: u_1 is assigned to place P_1, ucode: u_2 to place P_2,, and ucode: u_A to the relation that they are "adjacent." In this case, the ucR unit diagram representing that "place P_1 is adjacent to place P_2." shows that u_1 and u_2 are connected with u_A.

(2) Suppose that ucode: u_3 is assigned to a PET bottle of green tea which exists in the real world and ucode: u_B to the relation "name." The ucR unit

diagram indicating "the name of this PET bottle of green tea is ‘tasty green tea’." shows that u_3 and the atom that represents "Tasty Green Tea" are connected with u_B.

(3) Suppose ucode: u_C is assigned to the relation "instruction URL." The ucR unit diagram, which indicates "the instruction for the PET bottle of green tea described in the previous section is available at http://www.example.org/" associates u_3 and an atom indicating http://www.example.org/ is associated with u_C. Note that the URL indicating http://www.example.org/ is an atom, to which the ucode is not assigned.

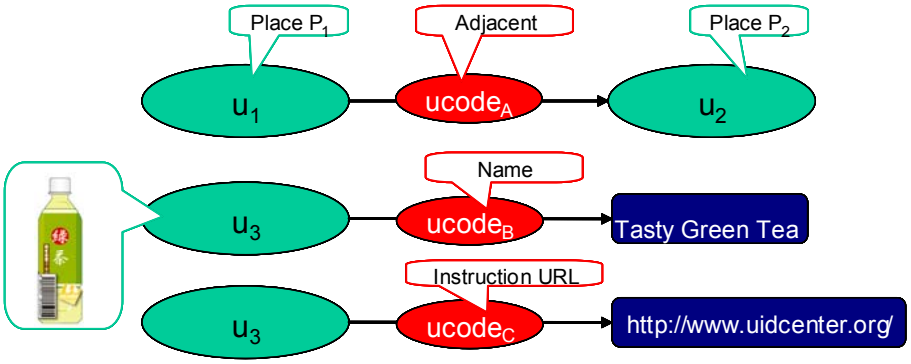

Figure 7: Example of ucR unit

3.1.2. ucR Graph

The information and context about an entity are represented by combining relations among multiple entities. For example, for information on an entity indicating a “stuffed cabbage,” pieces of information such as its recipe, a cabbage as ingredients, and a production area of the cabbage are combined. To represent such information, ucR units are combined, and a digraph where multiple ucodes and atoms are connecting by relation ucodes is built. The digraph generated based on the above method is called a **ucode Relation graph** (**ucR graph**). More specifically, the ucR graph is a massive graph structure, in which ucodes are connected by relations and pieces of information described as atoms are associated with each ucode. Figure 8 visualizes the ucR graph. Atoms always appear as leaves of the ucR graph.

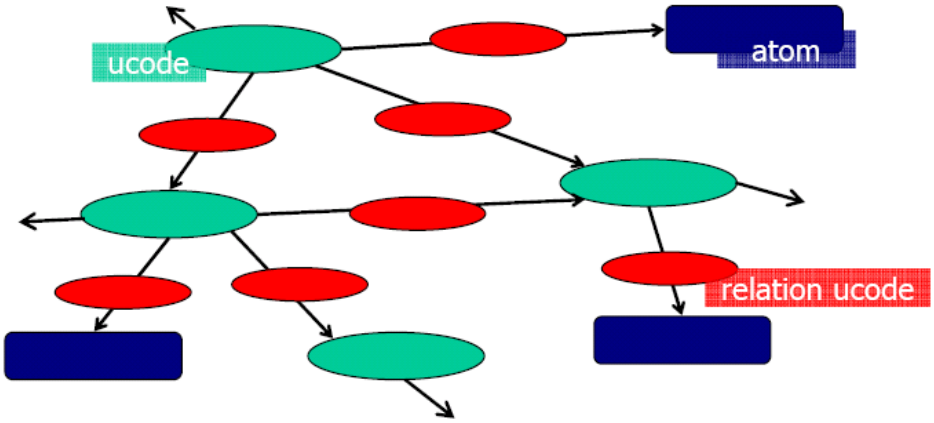

Figure 8: ucR Graph

For example, starting from a dish called a stuffed cabbage, as information on its ingredient, recipe, a production area of the ingredient, etc. is added, the ucR graph as shown in the following Figure 9 can be constructed.

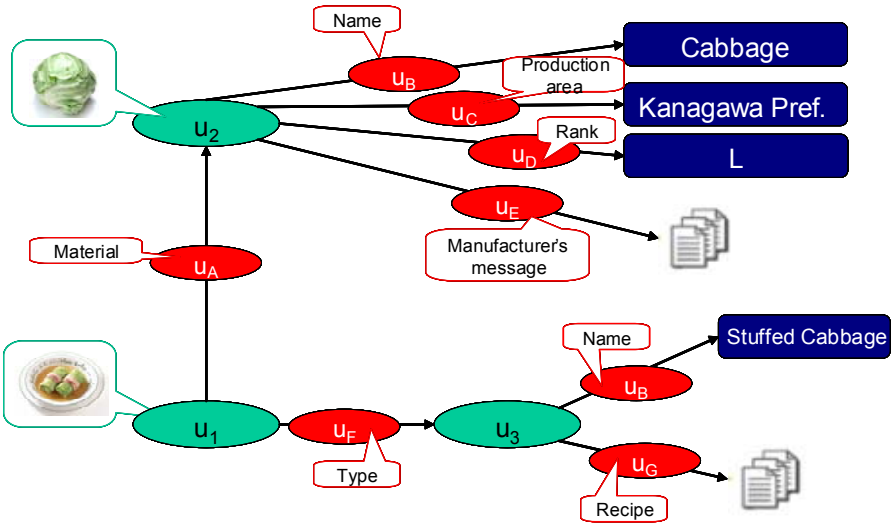

Figure 9: Example of a ucR Graph

3.2. ucR Format

A ucR format is a generic term of description specifications based on the ucR model.

There are three main ucR format specifications. The first stipulation is for stipulating the ucode representation method in a string text format (**serialized ucode**). The second is for serializing a ucR graph into a text or a binary format (**ucR format for Serialization**), and the third is for embedding the ucR format

for Serialization into content data in other formats (**ucR format for Content**).

3.3. ucR Vocabulary

The ucR graph indicates relationships among ucodes. The ucode itself is only a fixed-length value and does not essentially have any other meaning. On the other hand, if various types of applications are built on Ubiquitous ID architecture, the meaning of the ucR graph in each application should be understood by third parties. Therefore, it is necessary to know what logical ucodes mean in advance. For example, the food traceability systems recognizes ucodes indicating a grade and production area of crops, and the ubiquitous location information systems understand ucodes indicating that two places are adjacent, etc. The definition of the meaning of basic logical ucodes for which a common understanding is required for various applications is called Vocabulary..

The **ucR vocabulary** is a generic term that refers to vocabulary definition specifications.

In Ubiquitous ID architecture specifications, unless otherwise specified, "vocabulary" refers to the ucR vocabulary.

3.4. Simplified ucode Relation Model

In some applications, there may be only one type of relation between a subject ucode and an object ucode or between a subject ucode and an atom, or it may not be required to identify such relations. For those applications, Ubiquitous ID architecture permits omitting the relation ucode, recognizing there is an "implied relation" in the ucR unit. That is, the real-world context is represented by modeling information on entities in the real world as the implied relations between ucodes assigned to entities or between a ucode assigned to an entity and an atom. This representation model is called the **simplified ucode Relation model**.

4. ucode Resolution

This chapter outlines the mechanism for realizing the ucode model and ucR model consisting of three parts as follows.

- the ucode relation database,
- the ucode resolution which manipulates information using the ucode relation database, and
- the ucode information service which is obtained as a result of the ucode resolution.

This document describes an outline of these three mechanisms. Their details are defined in [3], [4], and [5].

4.1. ucode Resolution

4.1.1. ucR Database

The wide-area distributed database which manages ucR graphs is called a **ucode Relation database**. The ucR database comprehensively manages information on the relations among multiple ucodes in addition to the content such as information services associated with individual entities to which ucodes are assigned. The ucR database is basically an open database which anyone can use to reference or register information, but it can also implement an access control.

Ubiquitous ID architecture does not specify the internal configuration of the ucR database. However, the ucR database must provide an interface for ucode resolution explained in the following section.

4.1.2. ucode Resolution and ucode Information Service

When a user physically accesses an entity in the real world, Ubiquitous ID architecture identifies the information appropriate to the situation from the ucR database, based on the ucode assigned to the entity. This process is called **ucode resolution**. Moreover, the information associated with ucodes, namely, the ucR graph is registered in the ucR database. This process is called a **ucode registration**.

The protocol for accessing the ucR database in this manner is called **ucode Resolution Protocol** (ucodeRP).

The server, which provides ucode information services and content reached by performing the ucode resolution, is called a **ucode information server**.

The ucode resolution consists of four main function modules (Figure 10).

(1) ucR Database Node (UDN)

The **ucR database node (UDN)** is an individual node of the ucR database which executes distributed management of ucR units.

(2) ucR Database Front-End (UDF)

To search information on a certain ucode, it is necessary to build a ucR graph containing the searched ucode(s). However, since the UDN which manages ucR units is generally different depending on the ucode, it is necessary to collect necessary ucR units from distributed UDNs and build a ucR graph consisting of these units. The **ucR database front-end (UDF)** handles this process.

(3) ucR Vocabulary Engine (UVE)

Basically, the UDF builds a ucR graph, but it does not know the meaning of the ucR graph. A functional module that provides semantic inference and interpretation for the ucR graph obtained from the UDF to narrow down range of the intended information is called a **ucR Vocabulary Engine (UVE)**. The UVE may be implemented if the application-specific search logic is required for an application. For example, extracting location information from the ucR graph and searching for places within a specified range from a specified distance is one such application-specific UVE.

(4) ucode Information Service (ucodeIS)

The **ucode information service (ucodeIS)** is a service derived from ucR graph search results and used by users finally.

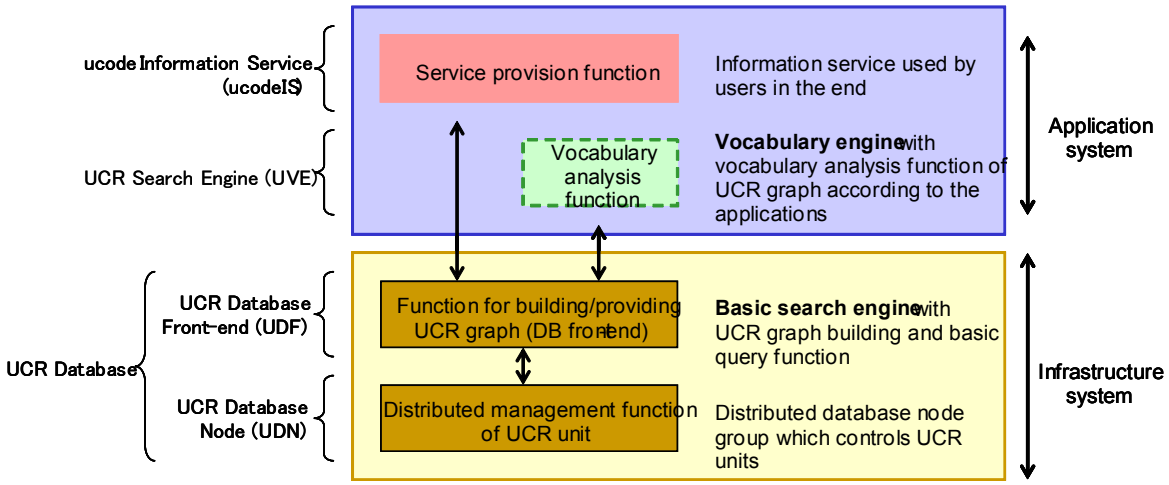

Figure 10: Basic Function Model of ucode Resolution

"(1) ucR Database Node" and "(2) ucR Database Front-End" of these four functional modules are part of Ubiquitous ID architecture, and "(3) ucR Vocabulary Engine" and "(4) ucode Information Service" are application-dependent.

In the actual systemconfiguration, each function module may be placed at any nodes in the system such as a server, a terminal, etc. Its placement pattern is not specified as part of ubiquitous ID architecture.

4.2. Simplified ucode Resolution

The ucode resolution performed based on the simplified ucR model is called **simplified ucode resolution**. The simplified ucode resolution is a directory service which identifies the address of content or services associated with ucodes.

The UDN in the simplified ucode resolution is a distributed database which manages the association between ucodes and the address of content or services, and this is specially called a **ucode resolution database**. In addition, the UDF in the simplified ucode resolution is a function module which identifies location of information on content or services associated with ucodes by querying one or more UDN. The server with a UDF function in the simplified ucode resolution scheme is specially called a **ucode resolution server**.

There is no UVE function in simplified ucode resolution. The server with a ucodeIS function in the simplified ucode resolution is sometimes called simply an **Information serveR.**

Index

V