TRONSHOW2014

u2
uID Architecture 2.0

Ken Sakamura
Director and Professor, Institute of Infrastructure Application of Ubiquitous Computing (IAUC), Interfaculty Initiative in Information Studies, The University of Tokyo
Director, YRP Ubiquitous Networking Laboratory
Chair, T-Engine Forum/uID Center
The IoT Model
Web Service Paradigm

Big Data

Input

Real World

Smartphone/Information Appliance
Operation, Video, Audio, Comment, Message

Display, Audio, Advice, Service

Output
Adding the IoT to Create the Total Paradigm

Big Data

Control and Maintenance Data

Output

Embedded Systems

Input

Data Measurement, reading IDs, Operation Data

Smartphone/Information Appliance

Real World
Input from the Real World as Big Data

Data from embedded sensor nodes
Operation data from embedded devices
Smartphone-based operation, video, photo, audio, comment and messages

These are collected and assembled over the network into Big Data
Output to the Real World using the Big Data

Efficient control of devices by analyzing Big Data

Fault prediction by analyzing Big Data

Improved maintenance and operation efficiency

Provide advice and services by using display and audio to human users
uID Architecture
The Essence of “Ubiquitous Computing”

Understanding of the situation / context of our living environment

- Location and spatial information of objects and people
- Attributes of objects and people
- Various sensor values

“Context Awareness”
What is Context-Awareness?

- Computer recognizing the status of the real world automatically
  - What is this?
  - Who is in this room?
  - Is this person visually-impaired?
  - Where is the device?
  - Who do you contact often?

- At the root of recognition is identification = IDentification
Assigning Identification Numbers to All which we want to recognize

The world consists of tangible objects and locations, but human activities in the world also involve notions or concepts

We assign numbers to such abstract concept as well
• E.g., a corporation, a lot, etc.
The Identification Number is ‘ucode’

128-bit ucode

• Can be unique to an object, location, or virtual object in an open and universal network
• A unique identification number in this world
Semantics externalized in the network

Note) Please do not confuse this with “Network externality” as in Economics
Merits of Externalizing the Semantics in the network

- Necessary resolution for application
  - An address can specify a building, but it can even specify a particular shelf within an office of the building if necessary

- Freedom for an application provider to issue an ID as necessary
  - It is possible to issue an ID on the spot when it becomes necessary

- Openness that allows ID to be used by anyone, not restricted to the issuer
  - Nevertheless, It can be declared private so that others can only tell that it has been issued
**Principle of Ubiquitous ID Architecture**

1. Obtain (1) ucode by automatic recognition of the ucode tag
2. ucode Tag
3. Ubiquitous Communicator
4. ucode Resolution Server
5. ucode Information Server

Public Network

Information/Service

Real World

Object (Product)

Place
2

Why a Number?
Why a “number” instead of a “name”?

● A “number” is a natural form for computers to handle
  ■ People are not good at handling numbers with many digits, and so use names instead

● A “number” can be used for unique identification
  ■ There are examples of different objects that share the same “name”
  ■ A “name” can not identify an instance of such similar objects
  ■ A proper name, when translated into foreign language, causes
Why a “number” and not a “name”?

- A “number” can be issued without special authority, as long as uniqueness is assured
  - When a “name” is used to identify something, we need to assign an “authority” to a party who can give such a “name”
  - When there are many things that are appropriate for a given “name”, who decides to give the name to a particular instance of them?
  - Can such a naming be agreed upon by the rest of the world?

- Full-featured management structure is not needed to assure uniqueness
  - Hierarchical naming such as URL and postal address reflects a particular management hierarchy, and if something is moved to a new hierarchy, a renaming has to occur, and thus perpetual identification is not assured
Social Significance of "Public Property" in the age of Open Data

The importance of assuring uniqueness only and having no meaning in itself
The Importance of Global Uniqueness

- The world is moving toward “open” systems
  - E.g., Open data, open government, …
  - The IoT (Internet of Things) can also be considered a huge open system

- In open systems, common rules are crucial for interoperability among data, services, etc.
  - E.g., Controlling KADENs (electronic appliances) in a smart house
    → Kadens in a house usually come from many manufacturers: we need globally unique identifiers for specifying each kaden

- "ucode" as the lingua franca for specifying objects and notions uniquely among systems
As a future foundation for metadata resilience

The persistence of IDs are crucial to assure the resilience of network of metadata of our society

- An identifier that belongs to a particular organizational domain, such as vendor model serial number, should be carried over by a more public database when the original issuer ceases to exist.

- To handle such merging of databases, an identifier system that has built-in semantics of a particular domain is not desirable.
Recommendations Based on ucode Have Been Consented at ITU (International Telecommunication Union)

  Definition of application requirements

  System architecture

  ID system architecture

- ITU-T H.642.2 (2012)  
  Process of ID registration and management

- ITU-T H.642.3 (2012)  
  Protocol for ID interpretation
International Standardization at IETF

- RFC 6588
  “A URN Namespace for ucode”
  - urn:ucode:_0123456789ABCDEF0123456789ABCDEF
  Representation above is standardized

- ucode can be used anywhere URI is used.
  - NFC (Near Field Communication) NDEF form
  - RDF (Resource Description Format) ID form
Applications of ucode

Applications that take advantage of inter-organization uniqueness of ucode

- Registration service of long live houses and housing components (Better Living)
- Pedigree Management of Horse Races (JAIRS, Japan Association for International Racing and Stud Book)
- Medical Drug Traceability System (Benesis)
- Intelligent Control Point (GSI, Geospatial Information Authority of Japan)
Research Projects on ucode

- R&D activities on foundation, and applications in Japan and overseas
  - Tokyo Ubiquitous Technology Project *kokosil*
    - Building Location-information infrastructure that uses ucode
  - Information distribution infrastructure (UNL/UCT)
    - Construction of open data platform based on ucode
  - IoT-A Project (EU FP7)
    - Led by VTT, EIT ICTLab, etc.
    - Adopted ucode as a part of resolution & discovery service in IoT-A architecture
3 ucode Relation Model
In order to provide information service that retrieves various relationships between objects and places of the real world,

A unique identifier (ucode) is assigned to things we want to identify
- ucode is standardized as ITU Recommendation

Real world (and virtual world) is modeled
- ucode Relation model is a framework to model the real world in a digital framework

Our “Ubiquitous Computing” concept is to utilize this model to optimally control the environment, while minimizing the explicit input of data
**ucode Relation Model (Definition)**

**Definition**

- A model to represent the context in the real world
- by representing the relationships between ucodes, or ucode and a literal (called `atom’)
- of ucode-identified objects, places and concepts

**ucR = ucode Relation**

- ucR model itself or a representation that uses ucR model are sometimes called ucR simply (informal usage)
**Example ucR (Object)**

12 boxes of Bufferin

- ucode₁
- ucode₂

ucode₂ contains ucode₃

ucode₃ contains ucode₁

ucode₁ is named as “Bufferin”

ucode₂ is named as “12 Bufferin boxes”
Example of ucR (Place)

\[ \text{ucode}_1 \quad \text{is close to} \quad \text{ucode}_4 \quad \rightarrow \quad \text{ucode}_2 \]

\[ \text{ucode}_2 \quad \text{is near} \quad \text{ucode}_5 \quad \rightarrow \quad \text{ucode}_3 \]
Example of ucR: A product and its parts

Contains

Shipment of a unit

Name

Date

2007/07/19
Important Points

- ucode permits a unique identification across boundaries of manufacturers and object categories
- ucode is never recycled: for one object, a unique ucode
- ucode can be issued anywhere by anyone: at manufacturer’s factory, on the usage spot, or by any business partners when authorized to do so
- Application provider can manage the access control what type of information can be retrieved by whom
  - E.g. For a part in a plant, it can be configured to allow maintenance personnel to retrieve the blueprint from the ucode while others may not even tell what it is, but may know that the part is used in a particular plant, or that it is used somewhere.
Important Points

- ucode is tag-agnostic, thus able to choose tags considering the cost and importance of items
  - QR Code seal for small low cost parts
  - Laser-etched QR Code for small expensive parts
  - A unit consisting of such parts above, it could be tagged with RFID tag that can be read in the presence of back scattering metal surface
    - If you purchase by a large volume, one is available at 100 YEN (about one US dollar) range.
  - The plants that have many such units can be covered by radio markers, etc.
Philosophical Underpinning of ucode

“Definition by Example” is the basic concept

- uCR uses the existing objects in the real world as its foundation
- ucodes assigned to objects and places in the real world are at the foundation

Abstract concepts such as relationships are deduced from the facts in the real world

- We do not need a prepared vocabulary in advance
**Limitation of RDF and ucR**

**RDF:** Tries to explain the real world from the abstract notion (top-down)

- http://www.example.org/stationid/85740
- http://www.example.org/terms/nextst
- http://www.example.org/stationid/85739
- http://www.example.org/terms/name

**ucR:** Defines the abstract notion starting from the real world

But "A map is not the real world itself."

---

Copyright © 2012 by T-Engine Forum, All Rights Reserved
A view: ucR is an RDF, but uses ucode instead of URI to specify the resources

RDF that needs vocabularies to model the real world is augmented by ucR
Combining uID and Smart-M3

- Globally unique identification + semantic operations

Cited from: Viljamaa, E et al, “A Smart Control System Solution Based on Semantic Web and uID,” 2011

SIB: Semantic Information Broker
KP: Knowledge Processor
uID Architecture: from U1 to U2
2.0 of TRON Project

To meet the new developments that make the IoT possible

1. Advances of technologies
2. Changes in the environment
3. Evolution of demands

T-Kernel has advanced to T2, and uID architecture to u2
1. Advances of Technologies

Distributed processing of Big Data

- MapReduce, Hadoop and other distributed data processing has become openly accessible

De facto standardization of open data processing protocol

- REST, CoAP, and JSON have become common
2. Changes in Environment

It has become common to create new services by using Cloud, mashups, and open data

- The age has come where a new service is provided by building it on top of others

Favorable environment for startups

- It is easy to use the advanced technology
  AWS (Amazon Web Services), GCP (Google Cloud Platform), etc.
3. **Evolution of Demands**

Demands from open data

• It is now crucial to automatically collaborate with the large open data sets on the Internet

Demands from Big Data usage

• It is now necessary to handle stream-type temporal context information for transportation open data, etc.
Example of Stream-type Context

Location of a Station
{STATIONuc₅, Ruc₆, geographical coordinates}

Interval/Lines and Stations.
{INTERVALuc₃, Ruc₇, STATIONuc₅}
{INTERVALuc₃, Ruc₈, STATIONuc₉}

{Station102, Ruc₆, 35.627163, 139.722698}

{IntervalF064, Ruc₇, Station102}
{IntervalF064, Ruc₇, Station103}

Static ucR

Overlaying Dynamic ucR...
Summarizing the “u1”

u1 has been a research project on model

- u1 successfully provided the information model, the unified data representation for the IoT
  - ucode for globally identifying objects/notions
  - ucR (ucode relation) for common representation of data

We now need practical implementations based on the idea of u1

- Must be implemented on the basis of current technology
Hurdles for Realizing IoT

In order for a successful deployment of IoT, many of the hurdles need to be overcome:

- Stability of uID services
  - Performance on high loads, 24/7 reliability, …

- Computational complexity and performance
  - NP hardness of subgraph matching, …

- Management complexities
  - How to manage billions of ucodes

- Protection against hostile behavior
  - DDoS attacks, access from malicious 3rd parties

“u2” aims at solving these practical issues
 Stability of uID Services
Affinity with latest cloud-computing technologies

- IaaS (Infrastructure as a Service), OpenStack, load balancers, ...
  - Amazon EC2, Google Compute Engine, ...

- u2 services are run on a IaaS service
  - Currently released to u2WG members
Computational Complexity and Performance
Large graph databases are intractable by nature

- Known from the theory of “computational complexity”
  - Because of the NP-completeness of subgraph isomorphism problem, graph queries are NP-hard
**Reasonable Restrictions on the query**

- Again, u2 is designed for practical usage
  - Although based on the u1 model, the aim of u2 is to realize a practical IoT information framework
  - Thus, we need to solve this problem

- u2 provides a novel solution to this problem
  - In u2, ucR graph and queries is not generalized to have an arbitrary form, but is restricted to meet several conditions
    - ucR graph is forced to have vocabulary-defined restrictions (e.g., “contains” predicate must form a tree)
    - Queries do not have wildcard predicates that match across arbitrary number of vocabulary sets
  - These reasonable restrictions help to realize practical yet flexible IoT information framework based on the ucR model
Basic Concept of U2

Information Infrastructure that permits cross-queries to various APIs of DBs across organization and DB boundaries

• uCR native database stores data that does not have a fixed set of queries, and is subject to complex queries. This part is implemented by RDF database consisting of ucode triples.
Use of Legacy DBMS

● Data with well-defined patterns and internal constraints are stored in legacy DBMS optimal for data schemas
  ■ DB domain is identified with the vocabulary group of the well-established relationship with internal constraints
  ■ Existing DBMS can be plugged into the whole system, as a virtual ucR DB, using a simple wrapper
  ■ The virtual ucR DB integrates other DBs such as KVS DBs, GIS DBs which use different schemas different from the native ucR.

● This makes it possible to use algorithm and data structure best suited to the nature of data and applications
  ■ Search convenience stores within 1km radius of the current position → GIS: R-tree is used.
  ■ Handling large volume of sensor data → KVS: use Distributed Hash Table (DHT)
Cross Query: Query Decomposition and Response Reconstruction

- Mobile Application
- ucR Query
- ucR Response
- Query Decomposition
- Response Reconstruction
- u2

Diagram showing connections between systems:
- RDB
- GIS
- RDF
- KVS

Copyright © 2012 by T-Engine Forum, All Rights Reserved
**ucodeRP2.0**

**ucode Resolution Protocol 2.0**

*Protocol to collect and integrate information necessary for the IoT*

- A design meant for data usage across domain boundaries by means of open model of ucR (ucode relation) model

  This allows the federation of legacy RDBs and KVS DBMSs and permits cross queries.
**ucode Resolution Protocol 2.0**

- ucode Resolution Protocol ver.2 uses standard technologies such as REST/CoAP, REST/HTTP, and JSON.

- ucR queries are decomposed for DB domains (corresponding to particular sets of vocabulary) and subqueries are processed in each domain.
  - ucR resolution server receives ucR query ad then interprets it, and decompose it into subqueries for each vocabulary group, and send each subquery to corresponding ucR adaptor.

- **Distributed Database with Hierarchy**
  - ucR Resolution server front-end is made of distributed database, and if the database doesn’t understand a vocabulary, it will send the query to a resolution database in the upper layer in the hierarchy for resolution.
  - For confidentiality, a local ucode resolution server can be set up in an organization. Then the whole steps of processing query and response can be performed locally.
ucodeRP2.0 Resolution Steps

(1) ucodeRP2.0 query

(2) Decomposed subquery A

(3) Result of a subquery

(4) The result to the original query by joining the results of subqueries

ucR Adaptor

RDBMS

Server for Company A

ucR Database (Triple-Based)

Server for Organization B

Load Balancer

ucR Resolution Server Front Ends

uID Center
Management Issues
Management of large number of ucode

How can we manage billions/trillions of ucodes?

- Basically, ucodes are never reused
  - Hence, management efforts are often unneeded
  - Stale ucode data can be removed from databases without affecting other portions of data

In u2, basic solutions for ucode management are provided

- Domicile ucR data service that can be used to define the basic information at the time of registration, is included in u2 service framework
  - E.g., owner info, context-aware access control information, …
- Providing REST API for facilitating the issuance of new ucode
Protection Against Hostile Behavior
Protection against Hostile Actions

● Authentication and access control are inevitable consequences of using the open IoT environment as application infrastructure
  ■ E.g.: Desire to permit only the people living in a house and who are in the house at the moment to control home electronics appliances (kaden).
  ■ E.g.: Allowing access to health or disability information only to specific services (such as kokosil barrier-free navigation)

● A platform consisting of three-legged OpenID authentication model and Context-based RBAC
  ■ We are carrying out the design and implementation of the platform and the authentication and access control platform for IoT nodes and 6LBR, which requires light-weight implementation.
DareSil

The platform for authentication and access control of IoT nodes and services based on u2
Layers in the Architecture

● Application Layer (equivalent to SaaS)
  ■ Self-contained services provided to end users
  ■ This layer consists of applications that are tailored specific services, and the content and vocabulary that go with it

● Platform Layer (equivalent to PaaS)
  ■ This layer provides special functions and libraries targeted to specific set of objects, locations, ad content, and specifically provides API to perform cross query over variety of data efficiently by way of uID Center
  ■ It defines and manages the attribute of “original data”（本籍データ） of ucode for each target data category

● Infrastructure Layer（equivalent to IaaS）
  ■ This performs the resolution according to ucodeRP2.0, and returns appropriate data
Layers in u2 Architecture
Five Application Platforms

● kokosil
  - A platform to manage the locations, and the relationship among them.
  - It handles the coordination with existing location-information and map-information services such as GIS.
  - It acts as the basic platform for location-information services, sensor network applications.

● monosil
  - A platform to manage tangible objects and the relationship among them.
  - It handles the coordination with existing asset management services that use ISBN, etc.
  - It acts as the basic platform for cross-organization traceability service, logistics, and manufacturing management, etc.

● kotosil
  - A platform to manage contents.
  - It handles the coordination with existing content sites such as Wikipedia, and Twitter.
  - It can act as the base of ucR-based content application and can work with kokosil to produce a guidebook application, or work with monosil to produce a catalog application.
  - kotosil can be called from other applications: e.g., kokosil can use the content related to a location.

● daresil
  - A platform to manage users, user groups, and organizations and the relationship among them.
  - It handles the coordination of authentication with existing SNS, and others.
  - This is used by other applications for user management and access control management.

● kachisil
  - A platform to manage accounts for fee-based services. This is used by other applications.
  - It handles the coordination with existing credit services.
⑤-1
Realizing the IoT by u2

*Future potential of u2 and open data*
**u2 for data governance**

**as infrastructure for open data**

An infrastructure to permits cross queries to variety of DBs across organizational and DB domain boundaries

- It accepts standard a ucR query from an application, decomposes the query into subqueries, sends subqueries to DBMSs where they are processed, constructs the response by integrating the responses, and returns it as the response to the original ucR query.
Future potential of u2 and open data

- Many data sets will become available as open data
- Their federated usage will be an issue
  - E.g. How to know the average value from all the sensors in an area
  - Obtain the list of buildings in the given area from open GIS,
  - Then, for each building in the list, invoke sensor API to collect data and record them, wait for the answers for all the queries, and once all the answers are in, then perform the statistical processing.
  - Such federated actions are difficult for mobile application with limited available resources
- If u2 handles cross queries and response construction within a single standard ucR query,
- varieties of applications can be built easily
Future potential of u2 and open API
High-speed, Low-cost and Always-on Network

Home electric and electronic appliances will always be connected with cloud

- For example, it is easier to conduct the collection and utilization of electricity usage at home by connecting all the home electric and electronic appliances with cloud instead of managing and analyzing the data by the installed server at home.

Embedded devices as “Bridges between cloud and the real world” in the IoT paradigm
Unbundling functions by OPEN API of Embedded Devices

Data processing by cloud
UI by smartphone
Specialized function by embedded device
u2 for the Governance of Control

Used as part of the upper layer of access control

Consolidation of information, context-awareness

Access control of devices and information
Access Patterns to open API of Embedded Systems

Via Web

• Accessing system-provided API
• Devices are accessed by means of vendor-provided web site for control and management

Local Direct

• Application issues API call directly to the device on the local network
Limiting Access Control through u2 Only

- **Via Web**
  - Integration and wrapping of OpenID-style authentication delegation of the original web API
  - We can access devices with very universal vocabulary which is converted to the each device’s API, so we can use the APIs from different vendors in a federated manner.

- **Local Direct**
  - U2 realizes the integration and wrapping of API and the input and retaining of access key
  - Access key is not exposed to user application, and so the embedded application does not have to handle complex access control only with the key.
As Access Control Infrastructure for Open API

Mobile Application

ucR Command

ucR Response

Command Decomposition

Response Reconstruction

VCR by B

TV by B

Web Site of C

Aircon by A

Washing Machine by B

Audio by B

Security Camera by C

Auto Lock by D

Fridge by A

Web site of E
u2 as open access infrastructure

Aggregation of information and context awareness

→ ucode Resolution Protocol (ucodeRP) 2.0

Access control management of devices and information

→ daresil
In the distant future, self-organizing world description

Generating “Predicate” from Big Data

• In an era when we can tell from statistics that A and B has a certain relationship, then it is desirable that the predicate for the relationship and the database domain that handles the related data are generated automatically for efficiency reasons.

The merit of ucode is that it can be easily assigned to something that is generated by application on the spot.

• It is not necessary to attach the meaning, “definition” and “name”, to such relationship.
TRONSHOW 2015

Dec 10 to Dec 12, 2014
TRON Project 30th year anniversary!
TRON Symposium

u2 API will be released.