C-004

A Gradable Electric Equipment based on Open Source Platform with Linked Open Data

Tadashi Ohashi[†1]

1. Introduction

An open source operating system such as Android has been popular today in the world. Under the circumstance, an electric equipment using embedded technology is able to dynamically be metamorphosed by means of Linked Open Data (LOD) and Field Programmable Logic Arrays (FPGA) embedded in the smart devices in which hardware`s information and embedded software products are registered and embedded software products are directly loaded into smart devices by means of LOD. Various embedded software products in the smart devices have types as well as versions. These types' and versions' information including object data are registered and widely allocated on the web which is comprised of LOD, which can take the role of Software Products Management on Web (SPMW) to gather various types of software products on the web. Therefore both Android [1] and FPGA as reconfigurable embedded system is well studied. This new type of smart device architecture called Embedded Architecture with Linked Data (eALD) is proposed, to be allowed to upgrade embedded systems which are matched hardware types and versions in accordance with user's desire or to downgrade embedded system which can maintain system sustainability. By the way, big problems have been laid. Reconfigurable information is dispersed in the world level. This paper consists of the following 5 items to solve problems.

(1) Current technologies and related problems of conventional embedded systems.(Chapter.2)
(2) How to solve these problems. (Chapter.3)
(3) New system as such as eALD system is proposed to solve current problems and future problems. (Chapter.4)
(4) Efficacy on proposed eALD system.(Chapter.5)
(5) Concluding remarks. (Chapter.6)

2. Current technologies and their problems

Today's hardware devices and modules are characterized by the following items.

2.1 History of electronic equipment with embedded technologies

Until 1980's, a computer consists of hardware and software. A computer with both hardware and software is developed through drawn circuits drafts on papers. A designer can complete designed system relying on design requirement specifications, detail design specifications and the like. For hardware and software design, documents are very important as mentioned in my papers [2) 3)].

2.2 Current embedded technologies

Nowadays embedded systems consist of the following 4 parts. This paper picks up a smartphone as an electric equipment. The following Category1 to 4 is abbreviated as (C1-C4).

§ C1.Hardware components

A usual embedded system is comprised of ARM (Advanced RISC Machines), CPU (Central Processing Unit), and GPU (Graphics Processing Unit), DSP (Digital Signal Processor), storage devices, sensors and display panel and so on. A hardware configuration varies from applied field to field. As for a smartphone, hardware configurations, types or versions are changing to update functions to be requested by producers or users and to downgrade for sustainability.

§ C2.Operating systems

There are various kinds of operating systems. Necessary requirements is multi-processing operating system such as Android operation system as well as iPhone. In this paper, Android is used because it is an open-source. Application products run under target smart devices or operating system.

§ C3.Firmware products (including emulator)

In the first stage, embedded software is called firmware to tell from conventional software. In this paper, terminology "embedded software" is not used but terminology "firmware" used instead. This category includes mainly hardware gate control software in storage devices. In the beginning, firmware is distributed as the form of external storage devices like PROM (Programmable ROM), PLD (Programmable Logic Device) and etcetera. And patched data is superseded by additional storage devices to exchange the revised content of firmware. Thanks for web technologies advancement, patch data or renewal of data are directly applied to customer's devices.

§ C4.Application products

Web browsers, address books, e-mailers and so on are included as application products. Generally all products are pre-installed. In additions, there are many products from App Store or Android Market (Google Play). These products are downloaded based on user's needs.

2.3 Related problems

Every development of hardware, operating system, firmware (emulator) and applications' development styles are impendent. This means very expensive cost and less reliability due to complexities. That's why eALD must be developed.

2.4 Problems of each category

For hardware components, real-time operating system, firmware's (emulator) and application, documents are very important. In addition, data are also very important but hardware and software are designed in separate divisions or companies and

†1 iLICT (http://www.ilict.jp/)

managed by different ways. As results, it is troublesome of supplying appropriate hardware, software and firmware from various kinds of servers of different divisions or companies. An embedded system production process is generally too much complicated due to 4 categories, as opposed to general ICT systems [4].

3. How to solve these problems

As mentioned above, In process of planning, designing, manufacturing, testing, maintaining, upgrading, and downgrading should be holistic approach. “Holistic” means totally practiced approach. It means that to make things simple. This approach makes troubles less in terms of hardware, emulator, firmware, applications. It makes maintained systems always latest versions. This is the biggest reason why eALD must be developed.

3.1 How to solve problems for related problems

To solve the above problems, the following items have been taken into account as for this paper’s research with reference to the surveyed paper [5].

(1) Dispersed resources of real-time operating system, firmware (emulator) and applications are holistically managed on cloud.

(2) Every resource has repository maid of RDF mentioned in the referred paper[6] or OWL (Web Ontology [7] [8] on the cloud.

(3) A given embedded system is called eALD system that can deal with the above (1) and (2).

(4) eALD is upgradable or down gradable computer such as configurable computer. Note that “upgradable” means reconfigurable to enroll many higher upgrading hardware, emulator, firmware, applications. “Down gradable” means reconfigurable to keep system sustainable.

3.2 Upgrading embedded software systems

(1) When a user decides upgrading, the eALD takes the next actions.

1) eALD checks current version of hardware components, operating system, firmware (emulator) and applications products.
2) eALD recognizes the operator’s request type and version of products.
3) eALD searches repository RDF through search engine eALDre and get LODs as mentioned in the document[1] .
4) Depending on LODs, eALD can upgrade embedded software products.

(2) When a user decides downgrading, the eALD takes the next actions.

1-3) are as same as upgrade.

4) As results, eALD can downgrade its system.

4. New system (eALD) is proposed

4.1 Survey of preceding study or research

Sometimes, “Reconfigurable” is very similar to “Virtual Machine” so that related paper[9] is listed in reference to this category.

Upgrading must be reconfigurable. So the survey on reconfigurable computing and the like have been conducted. Many related studies or researches have been made. Other related papers on Java are described in documents [10] [11]. Three preceding papers are listed-up as such FPGAs are used. With FPGA, a reconfigurable computing is possible, so that in this paper, FPGA involved study or research only is applied.

§ First preceding paper

The first paper [12] is very useful because three types of reconfigurable computing are studied. If hardware logic is fixed, then software takes the role of hardware. In this case cost and processing time are trouble, in many cases, processing speed generally become slowly if functions are first priority. As a result, trade-off of reconfigurable computing between hardware and software are important.

§ Second preceding paper

The study of second paper [13] deals with varieties of application environment. Four critical items of designing reconfigurable computing are studied. When eALD is developed, much attention must be paid for critical items. Maybe eALD development is progressed then, four critical items are very important.

A fixed-distribution dual-Vdd (Multi supply voltages) FPGA fabric which contains Many VddLs is not effective for eALD development. With respect to mulch stacked FPGAs are found in process of preceding study, this paper excluded these multi stack FPGAs for reconfigurable computing.

§ Third preceding paper

The third paper[14] is very useful in that (1) FPGA is used barreling with CPU. (2) This system superseded Java programming with hardware alternative operation to perform speedy. Java performance shows lot of hints for eALD. (3) This reconfigurable system shows also lot of hints for upgradable or down gradable eALD development.

For further improvements of the above written preceding research, Comments on eALD will be given to make an epoch for future convenient embedded systems.

4.2 Proposed system

To solve the above problems, eALD is proposed.

4.2.1 What`s eALD all about

Upgrading or downgrading must be reconfigurable, so that survey on configurable computing has been made. According to “Linked Data: Evolving the Web into a Global Data Space” advocated in page 7 of referred paper[1] must satisfy the following items.

(1) Use URIs (Uniform Resource Identifiers) as names for things.
(2) Use HTTP (Hypertext Transfer Protocol) URIs, so that people can look up those names.
(3) When someone looks up URI, provide useful information, using the standards (RDF[16], SPARQL[17]).
(4) Include links to other URIs, so that they can discover more

things.

"The basic idea of Linked Data is to apply the general architecture of the World Wide Web to the task of sharing. In order to understand these Linked Data principles, it is important to understand the architecture of the classic Web. The document Web is built on a small set of simple standards: URIs as globally unique identification mechanism, the HTTP as universal access mechanism, and HTML (Hypertext Markup Language) as a widely used content format. In addition, the Web is built on the idea of setting hyperlinks between Web documents that many reside on different Web services" as describe in Health's paper[15].

4.2.2 Software Products Management on Web (SPMW)

LOD has four ways of usage for SPMW which stands for Software Production Management on Web. As described in 2.2, each of category C1 to C4 has the common problem, namely each

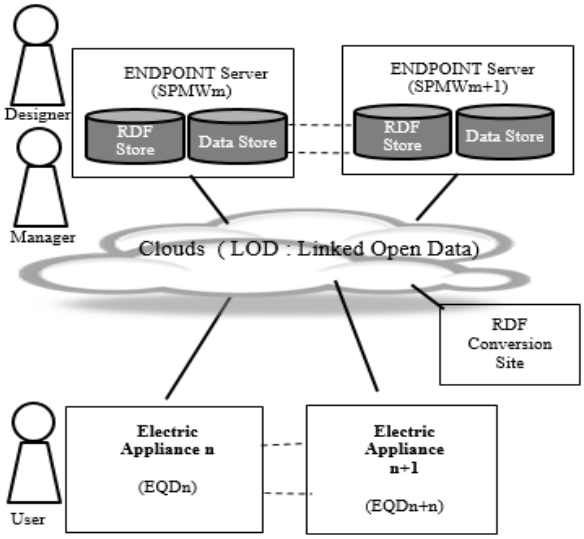

Figure1 Principle chart of Software Products Management on Web(SPMW) structured data on global scale.

Development and production process is not standardized, so that development and production management must be unified. For this reason SPMW is advocated.

Linked Data builds directly on Web architecture and applies this architecture to the task of sharing data on global scale. The first Linked Data principle advocates using URI references to identify, not just Web documents and digital content, but also real world objects (FPGA Data) and abstract concepts. " The above reference is conceptually exemplified in Figure1.

4.2.3 Fundamental concept on eALD

The meaning of eALD came from "yields" which mean that something will be produced from scratch. Based on given embedded system, a new function may be created by reconfiguration by hardware as an emulator, installing new firmware as well as applications. In this paper, term "upgrade" is used to distinguish from "reconfigurable" so as to escape the conventional meaning.

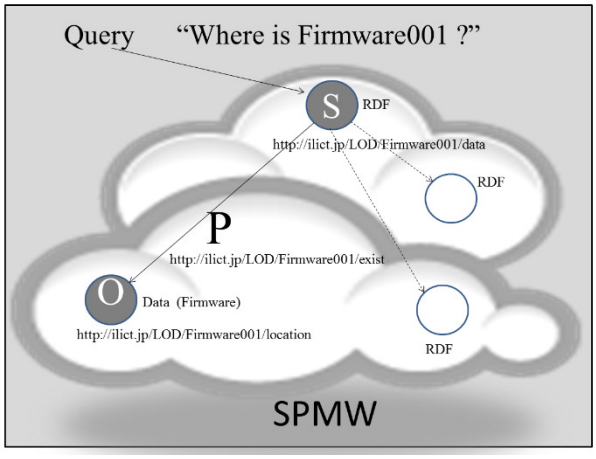

Figure 2 Principle diagram of Software Products Management on Web (SPMW)

Figure2 shows a kind of example of query issued by eALD's LOD retrieval engine (eALDre). A query is "where is Firmware001?" In order to denote, a sentence is given. A sentence is explained as semantic web as described in the referred paper [7] that S+P+O. S stand for Subject, P for Property, similarly O for Object. The answer is "Firmware001 exist on the server http://www.jp/LOD/Firmware001". This principle of LOD can expand to Figure2. Farther descriptions are 4.2.8.

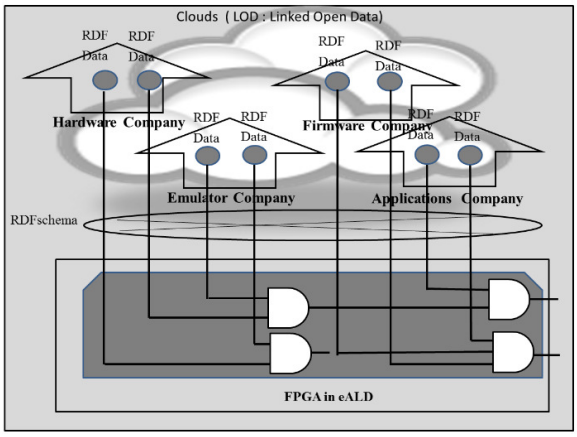

Figure 3 Fundamental concept on eALD

Fundamental Concept is shown in Figure 3 as SPMW. Upper part is cloud which holds LOD with RDF and Data. RDFschema as defined by [16] takes a role of spectacles. A mapping is performs between cloud and eALD. For an example, many logic gates in FPGA.

Each logic is performed by firmware based on VeriLog or VHDL as HDL (Hardware Description Language. Many electric circuits such as storages, ARM, DSP or etcetera are performed by HDLs. These components are software products which are delivered as electric components libraries on the web. Also these components are supplied by many different companies through the web. On the clouds, many companies deliverer their software products' element information (SPEI) or software products' group information (SPGI). SPEI and SPGI make software products enable to be accessed by user`s electric equipment such as smart devices. This fundamental idea shows that many

company can control many gates in FPGA directly though RDFschema for RDF as well as OWL.

4.2.4 Architecture of eALD

Figure4 shows substantial view of proposed system “eALD”. eALD consists of 5 major parts. 1) On Clouds, many LODs which are based on Semantic web[18] are prepared depending on company, products, as well as various versions of hardware emulator data, firmware, and applications. Further descriptions are discussed later. 2) eALDre is eALD’s search engine for RDFs on Open Linked Data. Main part of eALD system is the eACE (embedded Android Control Engine). eACE consist of ① embedded system (firmware) , ② Android OS, ③ Android Runtime which contains core libraries and DALVIK, ④Libraries, ⑤ Application framework, and ⑥ Applications. 3) Hardware consists of Analog devices, FPGA, Storage devices and sensors [19].

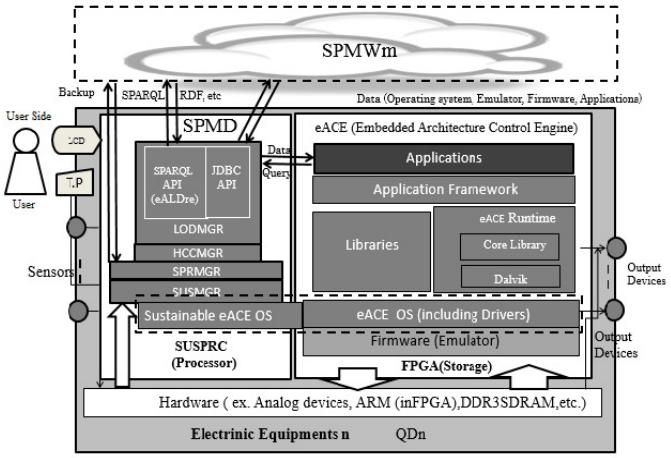

Figure 4 Gradable eALD architecture

4.2.5 Processing Sequence of eALD

With reference to Figure 4, first of all, SPMD (Software Product Management in Device) is detected both type and version. Secondly hardware type and versions are searched, and hardware’s emulators’ which take alternative software functions type and versions are searched. Firmware’s type and version are searched. Finally applications’ type and versions are searched. .For an example, a user wants version-ups for operating system. If there are request of Android operating system’s version from 2.1 (codename: Eclair) to 4.0.3 (codename: Ice Cream Sandwich). Procedures of this example are as follows.

(1) List-up all necessary hardware environments.
(2) Check current hardware environments.
(3) If not, find alternate hardware emulation. No alternate hardware emulators found. This user’s request is not feasible.
(4) If exist, search necessary firmware type and version.
(5) Search applications in accordance with operating system “Android OS 4.0.3”.
(6) List all of necessary and products for eALDre (eALD retrieval engines)

4.2.6 User’s required target system

Once the user’s required system’s target (Android operating system’s version from 2.1/API: 7 to 4.0.3/API:15) has been made in 4.2.5. Hardware is newly installed as example. Grade-ups are the operation system version 2.1 to 4.0.3. So that more hardware’s additional functions are expected. But as for example, only hardware emulator of new touch panel is newly graded up instead of hardware functions’ upgrading. Today`s technology requires additional hardware components but eALD may be able to substitute a necessary hardware with software products.so that hardware is not necessary.

4.2.7 eALDre retrieval engine

Once the user’s required system’s target has been made, eALDre is very important tool to find necessarily LOD based on the user’s required system’s target through for example SPARQL [17]. To make eALDre possible, it is truly depended on Tim Berners- Lee’s idea [20] as written in 4.2.3. Figure 2 shows Structural Linked Open Data for eALD.

4.2.8 Linked open data on cloud as SPMW

Linked Open Data is very important. Many companies must support products’ types and versions. From now on, many makers or companies support LOD. For experiment, LODs are supported by LOD site [21]. An excel sheets are prepared then, LOD site converts these excel sheets into RDF data. Corporations between LOD producers and LOD users must be corporate together.

Fig.5 shows fundamental software product ontology namely SPG I (Software Product Group Information) using PROTEGE Version 5.0 [22]. An every software product has SPEI (Software Product Element Information) such as Software Name, Version, Size, Data Location, and Performance Conditions of Current/Upgrade/Down Grade. Software Product Element Information for ontology (SPEIO) is comprised of Product Location of software product information and search result. Product information is comprised of Product ID (Product Name and Product Version), Product Type. This product type include Application, Firmware, Operating System and Firmware. And firmware is mainly FPGA data which is ARM control data, device control data, network control data, RAM control data, and ROM control data. In the past, ARM, device controller, network controller, RAM, ROM and other hardware components are comprised of hardware components. On the contrary, these hardware components are substituted by software in FPGA. By the way, search results are search state which means this product is in searching, and search compatibility is that this product is found on the web and compatibility check and performance condition are verified.

A searching element has upper version and lower version of the given software product. The Software Products are following four types. Each table is written in Excel sheets. Theses sheets are converted into RDF using Open Free Conversion Sites such as Google Refine[23] serviced by Google. Every SPEI consists of SPEIO, namely every software product has this SPEIO. An SPEIO is class which produces many objects like usual

programming. Thing[24] means class. Thing`s SPEIO is comprised based on RDF or OWL with RDF schema. Thing as class consists two major parts.

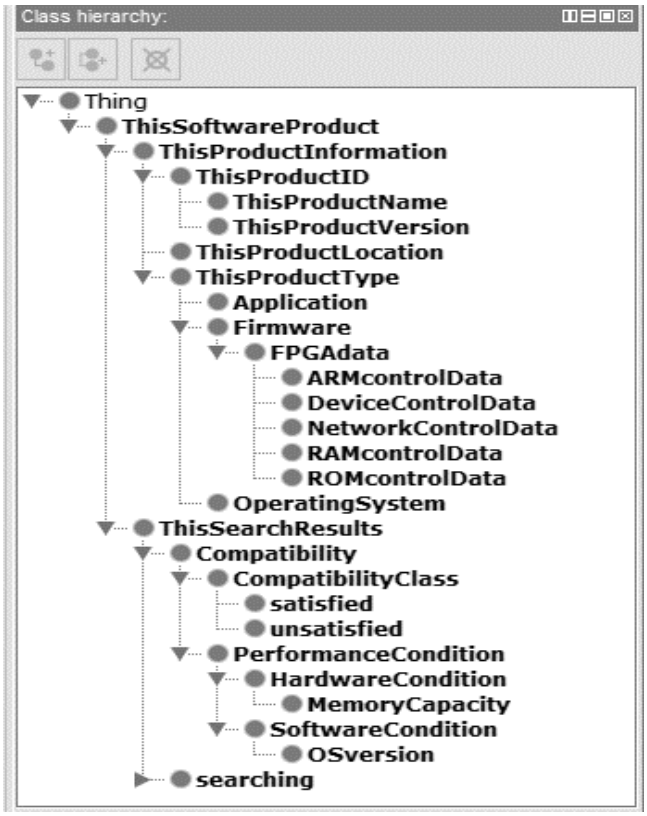

Figure 5 Software Product Element Information for ontology(SPEIO).

(1) This Software Products Information comprises of this products ID, product type and this product location. This product ID has this product name and this product version. This product Location is where software product object exists whose location is written in URL. Software product is object in the form of binary data. This product type has mainly three products. First of all is application, second product is firmware and last product is operating system. Firmware includes ARM control data, device control data, Network control data and RAM control data ROM organized in FPGA and control data written in VeriHDL or VHDL as HDL and loaded in to electric equipment's ROM or RAM organized in FPGA as binary data. (2) This Search Result comprises of compatibility and performance condition. A search results are condition after having searched for current version, upper version or lower version of a given software product. Compatibility means that under some conditions, searched software product is available or not. After having checked, compatibility result has two class levels. One of them is "satisfied". Another class is unsatisfied". If compatibility result is satisfied, an upgrade or downgrade is permissive. On the contrary, compatibility is not satisfied, an upgrade or downgrade is not arrowed. Another factor is checking of performance. This compatibility checks hardware environment. This checks whether memory capacity is enough or not. OS versions are fundamentally important software condition.

5. Gradable system process

Fig.6 shows that eALD process including SPMW and SPMD.A trigger is a left side upper arrow. A user select one of the menu where three select menu such as current search, upgrade operation or downgrade operation.

(1) Current search

In order to get current software product (a given software product name and version) element information. This menu is selected when a user makes it sure current status of user's electric equipment. The detail of current search process is not described here.

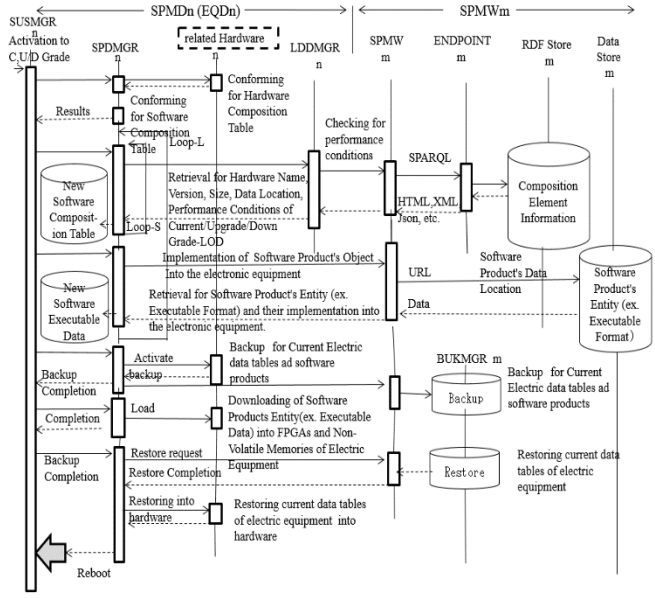

Figure 6 FPGA data sequence installed in eALD

(2) Upgrade and downgrade Operation

When a user wants the current electric equipment (EQn) `s SPEI , the user select upgrade or downgrade menu. This process starts with searching upgrade version's information as described in Fig.5 of given software product. Based on this upper or lower version's information, eALD searches for the upper or lower version SPEI using SPARQL. If the eALD found upper or lower grade SPEI through LOD, the eALD backs up current data and tables of a given electric equipment on SPMW. A necessary software products' objects are downloaded, SUSMGR loads them into FPGA, and in the consequence, restore backed-up data from SPMW into eALD storages. Finally SUSMGR reboot and eALD runs.

6. Effects on eALD

LOD is used on debug system or system generation process, in this aspect, there are no merits to enroll LOD on electric equipment or target smart device, but any application on for example App Store or the like is installed at the hand of users mobile smart system. For this reason, eALD equips eALDre to enroll LOD. From now, efficacy is described as follows.

(1)Independent developments of real-time operating system, firmware products (emulator) and application products can mash up on the web. As results, A proposed SPMW is effectively make it possible gather and manage the user's desired products. This is the most significant effectiveness of eALD. So that maintenance can be very effectively practiced.
(2) A user can upgrade software products or downgrade from smart devices as embedded Android system at hand by adding necessary hardware emulators without real hardware addition.
(3) If an application is developed, then it must be testing with all types and versions of hardware, but it is time consuming and expensive, by the way, this upgradable eALD makes it possible to debug new functions without necessary hardware.
(4) Sustainable manager (SUSMGR) is very important and useful for system's upgrading or downgrading.

7. Conclusion

eALD has been proposed. At this stage, this system is at cutting edge. But little by little, this eALD will be feasible.

7.1 What has been done

In this paper, the following items have been done.
(1) LODs on the SPMW make it clear to upgrade new system or downgrade sustainable system.
(2) Gate logics in FPGA of eALD may be controlled by LOD.
(3) Additive hardware may be possible to be performed by emulator.

7.2 Future study on proposed system

(1) Constraints on hardware's emulators exist. Lower level hardware may be in some cases difficult. For example, one CPU embedded system may not be 8 CPU smart system, in reverse case, 8 CPU smart system may be one CPU smart system may be possible. So that hardware's upgrade emulator must be studied more.
(2) An eALD system with LOD and FPGA will be more promising, functional, and important for the advent of the LOD era.
(3) Interpretation of RDF must be studied. This theme is very big. So that little by little, interpretation's engineering is advanced.
(4)How large can eALD deal logics in FPGA which is directly connected with URI through LOD. This theme is attractive because whether cloud can control FPGA gates directly or not.
(5) How to design sustainable system. Especially SUSMGR must be developed. Without electric power, without Android, upgrade or downgrade FPGA's object data must be swapped by current working FPGA object data.

References

1) Takashi Kishima, Sohei Ishimaru : Introduction to Android programming, *Magazine of Information Society of Japan*, Vol.52,No.4 and 5,pp.527-539 (2011)
2) Tadashi Ohashi: Reconsideration on Semantic web applied to Android System Design, *Android Bazaar & Conference 2013 Spring/Tokyo* (2013)
3) Tadashi Ohashi: Application of Web Linked Data to Android Embedded Systems' Design, *IPSJ Sig. EMB#29Joint Research Convention's Technical Report* (2013/5)
4)Shunsaku Egami, Hiroyasu Shimizu, Shota Taniguchi, and Akihiro Fujii : Mashup Applications Based on Screw LOD, *IEICE Technical Report,* AI2013-17,SC2013-11 (08-2013)
5) Masahide Kanzaki : Linked Data and Data Mapping, Journal of the Japanese Society for Artificial Intelligence,Vol.27 No.2,pp.163-170,Feburaly(2012)
6) Resource Description Framework (RDF)
http://www.w3.org/RDF/ (2004)
7) Vocabularies
http://www.w3.org/standards/semanticweb/ontology (2013)
8) Web Ontology Language (OWL)
http://www.w3.org/2004/OWL/ (2004)
9) Arnold, M., Fink, S.J. Grove, D., Hind, M. and Sweeney, P.F.: A Survey of Adaptive Optimization in Virtual Machine, in *Proc. IEEE*, Vol.93, No.2, pp.449-466 (2005)
10) Lattanzi, E. , Gayasen, A., Kandemir, M., Narayanan, V. Benini, L. and Bogliolo, A.: Improving java performance using dynamic method migration on fpgas, *Proc.18th International Parallel and Distributed Processing Symposium,* p.134 (2004)
11) Philip Faes, Mark Christians and, Dirk Stroobandt.: Interfacing java with reconfigurable hardware (2004).
12) Toshinori Sueyoshi, Masahiro Iida: Reconfigurable Computing, *Magazine of Information Society of Japan*, Vol. 40, No.8, pp.776-782, (August, 1999)
13) Philip Garcia, Katherine Compton, Michel Schaelte, Emily Blem, Wenyin Fu: An Overview of Configurable Hardware in Embedded Systems, *Hindawi Publishing Corporation EURASIP Journal on Embedded Systems*, Volume 2006, Article ID 56320, pp.1-19 DOI 10.1155/ES/2006/56320 (2006)
14) Keisuke Koike, Atsushi Ohta, Kohta Oshima, Kaori Fujinami, Nobuyuki Kohri, Masashi Takemoto, Hironori Nakajo : FPGA Acceleration of Java Applications in an Android, *Journal of Information Society of Japan,* Vol.53, No.12, pp.2740-2751 (Dec. 2012)
15) Tom Health, hristian Bizer: Linked Data: Evolving the Web into a Global Data Space, *Morgan & Claypool Publishers* (February 2011)
http://www.morganclaypool.com/
16) RDF Vocabulary Description Language 1.0 RDFSchema (2004)
http://www.w3.org/TR/rdf-schema/
17) SPARQL 1.1 Query Language
http:// www.w3.org/TR/2013/REC-sparql11-query-20130321/ (2013)
18) Semantic Web http://www.w3.org/standards/semanticweb/ (2013)
19) Shinich Nagano: Linked Data and Sensor Network, Journal of the Japanese Society for Artificial Intelligence, Vol.27 No.2, pp.200-206, February (2012)
20) Christian Bizer, Tom Health, Tim Berners-Lee: Linked Data, *Magazine of Information Society of Japan,* Vol. 52, No.3, pp.284-292, (2011)
21) http://linkdata.org/ (2013)
22) Protege Version 5.0 http://protege.stanford.edu/
23) Google Refine https://code.google.com/p/google-refine/
24) http://www.slideshare.net/yayamamo/7-linked-data-yayamamo