Methods for System Sustainability of Embedded Android Devices with Linked Data

Tadashi Ohashi[†1]

Smart devices are very high-performance so that system sustainability tends to be downgraded. Once a user get the system, it is very difficult to catch up with latest version of hardware functions and software products from marketplaces. To solve the problem, new methods must be applied to smart devices which are able to access dynamically to hardware functions in the form of emulator and software products through web Linked Open Data (LOD). The proposed methods make it possible to accommodate with the latest version of hardware functions superseded by emulators and software products which are directly loaded into storage component like FPGA by means of web LOD. The system sustainability of a proposed embedded Android device with Linked Data (eALD) can be performed by Sustainability Manager which deals with operating system, emulator, firmware and applications with results of software management information. LOD Manager downloads upgradable software products based on endpoint result of locations and Software Product Manager stores them into storage components of eALD from data store server of software download site. As results, eALD can perform its system's sustainability.

1. Introduction

Smart devices such as smartphone e.g. iPhone or Android are very important underpinning of today`s social system. But system sustainability tends to be downgraded. Once a user get the system, it is very difficult to catch up with latest version of hardware functions and software products from marketplaces. By the way, big problems have been laid for system upgrade to perform system sustainability. The technical information of reconfigurable computing is dispersed in the world-wide level.

To solve the problem, new methods must be applied to smart devices which are able to access dynamically to hardware functions in the form of emulator and software products through web Linked Open Data (LOD). The proposed methods make it possible to accommodate with the latest version of hardware functions superseded by emulators and software products which are directly loaded into storage component like FPGA (Field Programmable Gate Allay)by means of web LOD.

The reconfigurable embedded systems are little by little progressed owing to FPGA installed in Android smartphone. An open-source and real time operating system such as Android has been popular today in the world. Therefore both Android[1)] and FPGA as reconfigurable embedded system are well studied. The reconfigurable device means that system upgrade can be performed easily.

To make this method practical, an embedded Android device with Linked Data (eALD) is proposed. As results, eALD can perform its system's sustainability.

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 Historical background of current 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[2)3)].

2.2 Current technologies

Nowadays embedded systems consist of the following 4 parts. Each paper picks up a smartphone as an example. The following Category1 to 4 (C1-C4) is accommodated to the left vertical cells in Table1.

2.2.1 Hardware components

A usual embedded system is comprised of ARM (Advanced RISC Machines), CPU (Central Processing Unit), GPU (Graphics Processing Unit), DSP (Digital Signal Processor), storage devices like DDR3 SDRAM (Double-Data-Rate3 Synchronous Dynamic Random Access Memory), 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 upgrade functions to be requested by users.

2.2.2 Operating systems

There are various kinds of operating systems. Necessary requirement is a real-time and multi-processing operating system such as iPhone operation system as well as Android. In this paper, Android is used because it is an open-source. Application products run under target smart devices like Android smartphone. Generally operating system comes under

†1 iLICT Co. (Homepage http://www.ilict.jp/)

categories of firmware, for this paper, it becomes independent of firmware category.

2.2.3 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 machines.

2.2.4 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.

Table1 Related problems for embedded systems

Categories		Products, Components	Revision Control of products	Source Management for products
C1.	Hardware	ARC,CPU,GPU,FPGA,PDL, GPS, DDR3SDRAM ,DSP,LCD,	Product revisions are maintained by every section or company and indipendent of other products	Sources are managed by different computers in every other sections or companies but indipendent of other products
C2.	Operating system	An open-source Android OS		
C3.	Firmware	Gate control Software		
	Emulator	Emulators are partially or totally substituted by alternetives software products.		
C4.	Applications	Applications run under Android os		

2.3 Related problems

Table1 is shown as problems of the above described "2.1 Historical background of current technologies and 2.2 current technologies". The left cells on Table1 are written as categories. All four cells problems (C1-C4) are independent management style. 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 categories

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 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]. Problems are partly because of less sustainable system.

3. How to solve these problems

As mentioned above, In process of planning, designing, manufacturing, testing, maintaining, and upgrading 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. So that system sustainability can't be performed.

3.1 How to solve problems for related items

To solve the above problems, the following items have been taken into account as for this paper's research with reference to Linked Data and Data Mapping[5].

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

②Every resource has repository maid of RDF[6]with RDSs[7] or OWL[8] (Web Ontology[9] of Language) on the cloud.

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

④ eALD is upgradable computer such as reconfigurable computer. Note that "upgradable" means reconfigurable to enroll many higher upgrading hardware, emulator, firmware and applications to sustain system.

3.2 Upgrading embedded software systems

When an operator 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[5)10)].
4) Depending on LODs, eALD can download embedded software products.
5) eALDs can be upgradable and sustainable for 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[11] is listed as shown in Table2. Upgrading must be reconfigurable. So the survey on reconfiguring computing and the like have been conducted. Many related studies or researches have been made.

Reconfigurable researches are as shown in Table2. Other related papers on Java are described[12) 13)]. 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.

Table 2 Preceding survey of study of research

	Preceding paper 1	Preceding paper 2	Preceding paper 3
Title	Reconfigurable Computing	An overview of Reconfigurable Hardware in Embedded System	FPGA Accleration of Java Application in an Android.
Issued	1999	2006	2012
Written by	Toshinori Sueyoshi et al.	Philip Garcia et al.	Keisuke Koike et al.
Architecture	①Atached Processor ②Reconfigurable Coprocessor ③Reconfigurable Processor	Fixed distribution and part of vdd FPGA fabrics	FPGA and CPU used. Java source code runs on FPGA. DalvikVM is operation on Intel Atom Processor.
Merits	①Take alternate part of hostcomputer ②Combination with general micro processor ③Special part of host processor	①Lower Power ②Fault Torelant System ③Real Time Support ④Design Security	①Reconfigurable Android evaluation environment constructed. ②Java Applications run faster than ever before. ③Android can be controlled by FPGA as first case in the world.
Demerits	①Device problems ②Development problems ③Application problems	Nothing	This paper didn't include process time for configuration by JTAG. So that evaluation of this equipment is little doubtful.
Comments	This paper is not reserch but study.	Reconfigurable Computing	This paper is reserch. In this paper, Many referrences of reconfigurable computing.

Note that if each cell has items entitled by ①.②.③,these entitle numbers are related on vertical cells only. Namely within paper, entitle numbers are relative.

4.1.1 First preceding paper

The paper[14)] 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 is important.

4.1.2 Second preceding paper

The study[15)] deals with varieties of application environment. Four critical items of designing reconfigurable computing are studied as shown in Table2. When eALD is developed, much attention must be paid for critical items. May be eALD development is progressed then, four critical items are very important.

A fixed-distribution dual-Vdd 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.

4.1.3 Third preceding paper

The paper[16)] is very useful in that (1) FPGA is used barreling with CPU. (2) This system superseded Java programming with hardware alternative operation to perform lastly. Java performance shows lot of hints for eALD. (3) This reconfigurable system shows lot of hints for upgradable eALD development else.

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 based on Table2 to sustain the system performance.

4.2.1 Relation between Linked Data and eALD

Upgrading must be reconfigurable and sustainable, so that survey on configurable has been made. According to “Linked Data: Evolving the Web into a Global Data Space” advocated in page 7 of Health’s reference[10)], LOD 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, SPAQL[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 structured data on global scale. 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. Many hyper documents reside in different web services”. [10)]

4.2.2 An expanded idea applied to eALD

The above relation may applied to eALD as shown in Fig.1

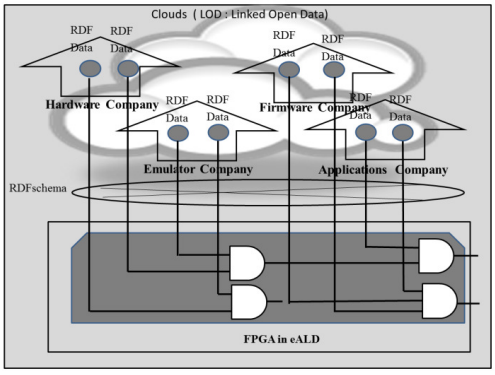
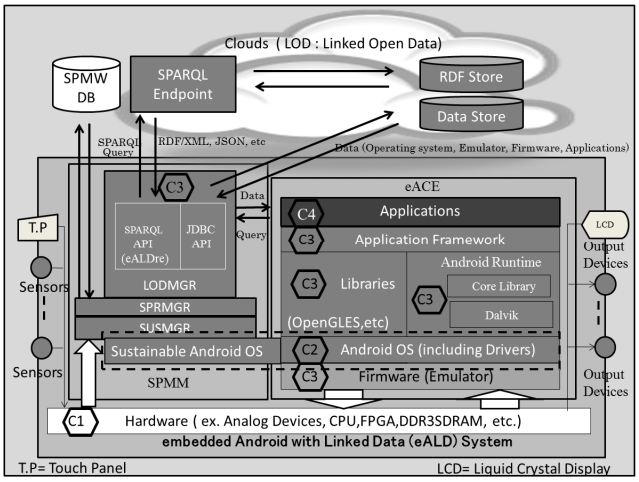


Fig.1 Fundamental concept on eALD

Upper part is cloud which holds LOD with RDF and Data. RDFschema[7] takes a role of spectacles that. data mapping is performed between cloud and eALDS.

4.2.3 Architecture of 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.



Note that C1 to C4 are categories as described in 2.3

Fig.2 Upgradable eALD architecture

Fig.2 shows birds 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 ①embedded system (firmware) , ② Android OS, ③ Android Runtime which contains core libraries and DALVIK, ④ Libraries, ⑤ Application framework, and ⑥Applications.

(3) Hardware consists of Analog devices, CPU, FPGA, storage devices like DDR3 SDRAM and sensors[19].

4.2.4 Software Products Management on Web (SPMW)

LOD has four ways of usage for SPMW which stands for Software Production Management on Web. As shown in Table1, Each of categories C1 to C4 has the common problem, namely each 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 and abstract concepts. " The above reference is conceptually exemplified in Fig.3.

Fig.3 shows a kind of example of query issued by eALD's LOD retrieval engine (eALDre) as shown in Fig.3. A query is "where is Firmdata006?" In order to denote, a sentence is given. A sentence is explained as semantic web[18] that S+P+O. S stand for Subject, P for Property, similarly O for Object. The answer is "Firmdata006 exist on the server http://spmw.org/lod/firmware#gate_control6_0101" as shown in Table5. Farther descriptions are 4.2.8. This principle of LOD emanated from Fig.1.

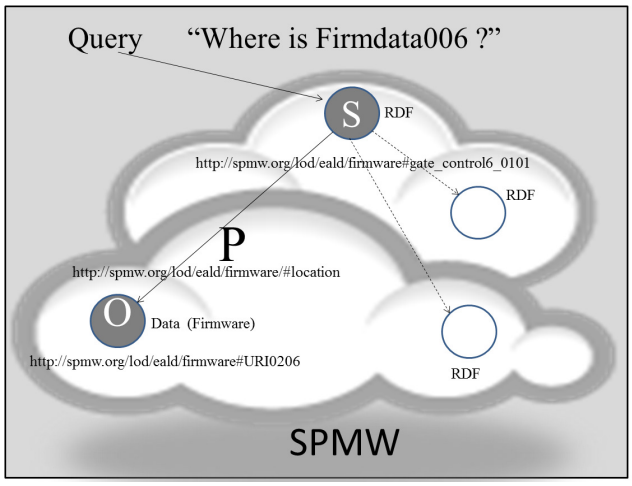


Fig.3 Principle chart of Software Products Management on Web (SPMW)

4.2.5 Sequence chart of eALD

The two major sequences are shown in Fig.4 and Fig.5. Fig.4 shows SPMM process and SPMW process. SPMM process consists of SUSMGR, SPRMGR and LODMGR.

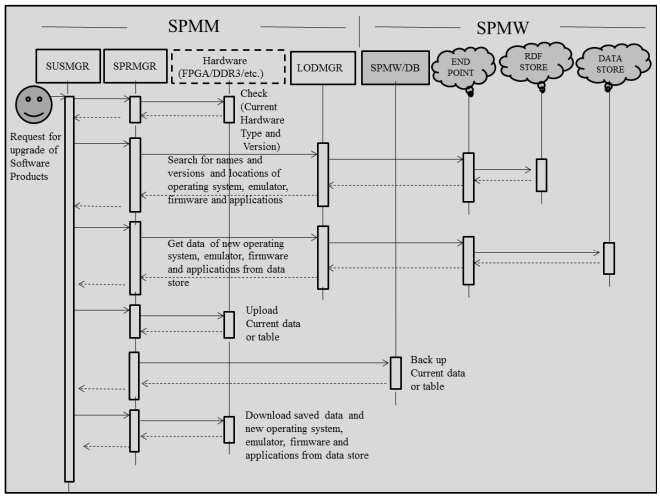


Fig. 4 searching for new software products by SPMM and SPMW

(1) SUSMGR is very important process as follows.

① SUSMGR controls total process.

② SUSMGR detects current hardware component names and versions through SPRMGR.
③ SUSMGR displays the process of eALD to user. Each process points are shown in LCD.
④ If a user recognizes the normal process, the user touches the panel that indicate the recognition button,
⑤ SUSMGR activates lower level SPRMGR.
If SUSMGER completes grade-up, it makes eALD starts reboot.

(2) SPRMGR activate LODMGR to retrieve LODs.
(3) LODMGR retrieves LODs by SPARQL through API (Application Programming Interface).
(4) SPMW consists of ENDPOINT process, RDF store and Data store.

Fig.5 shows that Currently Installed Products are searched. First of all, Operating system 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.

At this stage, 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.

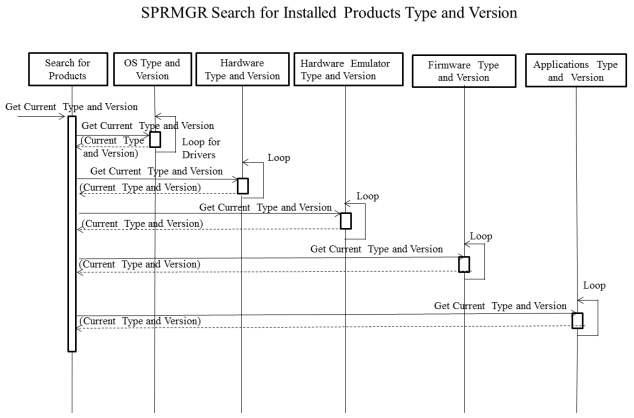


Fig.5 searching for new software products by SPRMGR

(5) Search applications in accordance with operating system "Android OS 4.0.3".
(6) List all of necessary and products for eALDre (eALD re 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. In this center part of Fig.6, 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.

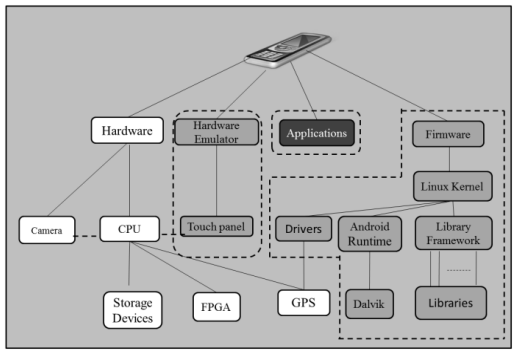


Fig. 6 The user's required target system with hardware and software (firmware)

4.2.7 eALD 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 SPAQL[17]. To make eALDre possible, it is truly depended on Tim Berners- Lee's idea[20] as written in 4.2.1. Fig.6 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 (Table3-Table6) are prepared then, LOD site converts these excel sheets into RDF data[21]. Table3 to 6 shows this excel example. Corporations between LOD producers and LOD users must be corporate together.

There are four experimental data. And C1 to C4 are categories as described in 2.2 and as shown in Fig.7.

(1) Experimental operating system ・・・・・・・・・ [Table3].
(2) Experimental hardware emulator ・・・・・・・・・ [Table4].
(3) Experimental firmware・・・・ ・・・・・・・・・ [Table5].
(4) Experimental application ・・・・・・・・・・・ [Table6].

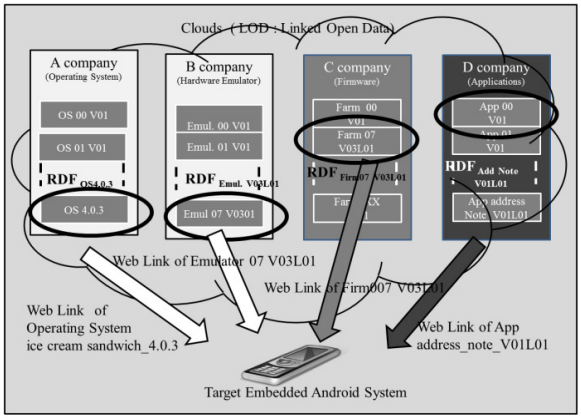


Fig.7 Linked Open Data for eALD.

C1. Operating system table (OST)

As for experimental operating system table is not exist, so that to make LOD, Linked Open Table for OS is made at hand as shown in Table3. In this excel sheet, tentative operation system numbers are shown on the left column. Software code name is on the next column. Software version and API level are shown. A tentative download site is designated. URI shows where an operating system exist on[5].

Table 3 Example of Linked Open Data Table for OS(OST)

Software Product No.	Software Code Name (Software Type)	Software Version	API Level	Software Product Source Location*
Operation System 00	Eclair	2.1	7	URI0000
Operation System 01	Froyo	2.2	8	URI0001
Operation System 02	Gingerbread	2.3	9	URI0002
Operation System 03	Gingerbread	2.3.3	10	URI0003
Operation System 04	Honeycomb	3.0	11	URI0004
Operation System 05	Honeycomb	3.1	12	URI0005
Operation System 06	Honeycomb	3.2.X	13	URI0006
Operation System 07	Ice Cream Sandwich	4.0.1	14	URI0007
Operation System 08	Ice Cream Sandwich	4.0.3	15	URI0008
Operation System 09	Jelly Bean	4.1.X	16	URI0009
Operation System 10	Jelly Bean	4.2.X	17	URI0010

* For example of URI, http://spmw.org/lod/eald/os#icecreamsandwich_4.0.3
Note that the domain (spmw.org) is not exist at this time.

C2. Hardware emulator table (HET)

Table 4 Example of Linked Open Data Table for Hardware Emulator (HET).

Software Product No.	Alternative Hardware functions (Software Type)	Software Version	Software Product Source Location*
Emulator00	Hardware00	V01L01	URI0100
Emulator01	Hardware01	V01L01	URI0101
Emulator02	Hardware02	V01L01	URI0102
Emulator03	Hardware03	V03L01	URI0103
Emulator04	Hardware04	V01L01	URI0104
Emulator05	Hardware05	V01L01	URI0105
Emulator06	Hardware06	V01L01	URI0106
Emulator07	Hardware07	V03L01	URI0107
Emulator08	Hardware08	V01L01	URI0108
Emulator09	Hardware09	V01L01	URI0109
Emulator10	Hardware10	V01L01	URI0110

* For example of URI,
http://spmw.org/lod/eald/emulator#hardware07-0301
Note that the domain (spmw.org) is not exist at this time.

Linked Open Data Table for Emulator must be similarly made. At this stage, it is doubtful how much hardware devices' emulators are feasible.

But temporally table is made as shown in Table4. Tentative emulator`s type is software product No. on the left column, tentative hardware type is on alternative hardware functions column, Emulator's version is on the software version column, emulators' source data location is the right column as URI.

C3. Firmware Table (FWT)

Operating system such as Android should be preliminary contained in this firmware category, but it is special product. So that it is excluded in this category. In Table 5, firmware's number is on the software product No. column, tentative firmware type is on the hardware control function column, Firmware's version is on the software version column, firmware' source data location is the right column as URI.

Table 5 Example of Linked Open Data Table for Firmware(FWT).

Software Product No.	Hardware Control Functions (Software Type)	Software Version	Software Product Source Location*
Firmdata000	gata_control0	V01L01	URI0200
Firmdata001	gata_control1	V01L01	URI0201
Firmdata002	gata_control2	V01L01	URI0202
Firmdata003	gata_control3	V03L01	URI0203
Firmdata004	gata_control4	V01L01	URI0204
Firmdata005	gata_control5	V01L01	URI0205
Firmdata006	gata_control6	V01L01	URI0206
Firmdata007	gata_control7	V03L01	URI0207
Firmdata008	gata_control8	V01L01	URI0208
Firmdata009	gata_control9	V01L01	URI0209
Firmdata010	gata_contro1l0	V01L01	URI0210

* For example of URI,
http://spmw.org/lod/eald/firmware#gate_control6_0101
Note that the domain (spmw.org) is not exist at this time.

C4. Application Table (APT)

Table6 Example of Linked Open Data Table for Applications (APT).

Software Product No.	Applications (Software Type)	Software Version	Software Product Source Location*
Application 00	Web Browser	V01L01	URI0300
Application 01	E mailler	V01L01	URI0301
Application 02	Callendar	V01L01	URI0302
Application 03	Map	V01L01	URI0303
Application 04	Address Note	V01L01	URI0304
Application 05	Mouse support	V01L01	URI0305
Application 06	Software Keyboard	V01L01	URI0306
Application 07	Game00	V01L01	URI0307
Application 08	Game01	V01L01	URI0308
Application 09	Game02	V01L01	URI0309
Application 10	Game03	V01L01	URI0310

* For example of URI,
http://spmw.org/lod/eald/application#address_note_0101
Note that the domain (spmw.org) is not exist at this time.

Table6 shows applications which run under operating system. Tentative application`s number is on the software product No. column, application name is on the software type column, application's version is on the software version column, application' source data location is the software product source location column as URI.

The Excel sheets of Table3-6 are converted into RDF as LOD using RDF conversion site or tools[21]. At this stage, structural trees are nor shown, but in some cases, it is possible. As results this SPMW performs sustainable software system as shown Fig.8 as such operating system is a basic transition role.

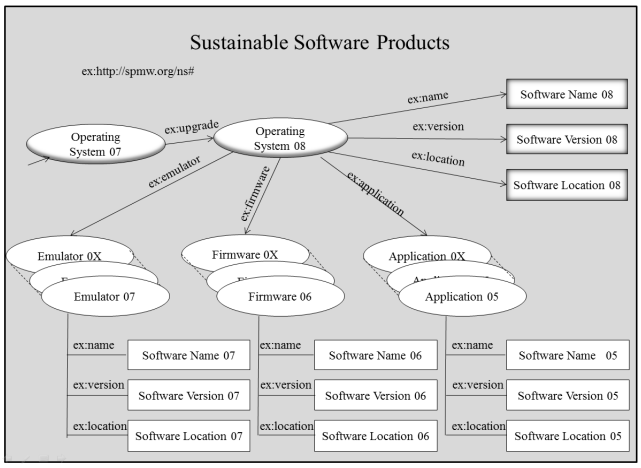


Fig. 8 Sustainable Software Products

5. Efficacy on New System

LOD is used on debug system or system generation process, in this aspect, there are no merits to enroll LOD on target smart device or embedded system, but any application on for example App Store or the like is installed at the hand of users mobile embedded system. For this reason, eALDS 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 gether and manage the user`s desired products. This is the most significant effectiveness of eALDS. So that maintenance can be very effectively practiced.

(2) A user can upgrade 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.

6. Conclusions

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

6.1 What has been done

In this paper, the following items have been done.

(1) LODs on the SPMW and SPMM make it clear to be an upgradable new system to keep sustainability.

(2) Gate logics in FPGA of eALD may be controlled by LOD.

(3) Additive hardware may be possible to be performed by emulator.

6.2 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 embedded system, in reverse case, 8 CPU embedded system may be one CPU embedded 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.

Reference

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, J*ournal 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) RDF Vocabulary Description Language 1.0 RDFSchema (2004)
http://www.w3.org/TR/rdf-schema/
8) Web Ontology Language (OWL)
http://www.w3.org/2004/OWL/ (2004)
9)Vocabularies
http://www.w3.org/standards/semanticweb/ontology (2013)
10)Tom Health, hristian Bizer: Linked Data: Evolving the Web into a Global Data Space, Morgan & Claypool Publishers (February 2011)
http://www.morganclaypool.com/
11) 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)
12)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)
13)Philip Faes, Mark Christians and, Dirk Stroobandt.: Interfacing java with reconfigurable hardware (2004).
http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.124.7704&rep=rep1&type=pdf
14)Toshinori Sueyoshi, Masahiro Iida: Reconfigurable Computing, *Magazine of Information Society of Japan*, Vol. 40, No.8, pp.776-782, (August, 1999)
15)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)
16)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)
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,Feburaly(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)