한국산학기술학회논문지
Vol. 12, No. 4 pp. 1844-1866, 2011

A Design Technique of Configurable Framework for Home Network Systems

Chul-Jin Kim[1], Eun-Sook Cho[2*] and Chee Yang Song[3]

[1]**Dept. of Computer System, Inha Technical College**

[2]**Dept. of Computer Software, Seoil University**

[3]**Dept. of Software, Kyungbuk University**

홈 네트워크 시스템을 위한 재구성 프레임워크 설계 기법

김철진[1], 조은숙[2*], 송치양[3]

[1]인하공업전문대학 컴퓨터시스템과

[2]서일대학 컴퓨터 소프트웨어과

[3]경북대학교 소프트웨어과

Abstract In a home network system, each customer electronic device has the control data format chosen by its manufacturing company and there are various types of digital devices and protocols. Besides the mutual operating environments among the various devices are dissimilar. Affected by the characteristics explained above, home network systems can hardly support the crucial functions, such as data compatibility, concurrency control, and dynamic plug-in. Thus, the home network system shows relatively poor reusability. In this paper, we suggest design technique of configurable framework, which can widely support the variability, to increase the reusability of the home network system. We extract the different parts of the home network system as variation points, and define them as the variability types. We design a structure of configurable framework, and suggest customization technique of configurable framework through selection technique and plug-in technique. Also, we prove the reusability by applying the proposed framework and it methods to real-world home network systems and analyzing the measurement results of these case studies using software metrics. We can expect the proposed approach provides better reusability than the existing them by analyzing those measurement results.

요 약 홈 네트워크 시스템에서는 각각의 전자 디바이스는 제조사별로 부과된 고유한 제어 데이터 포맷들을 가지고 있으며 거기엔 다양한 디지털 디바이스와 프로토콜 타입들이 있다. 게다가 다양한 디바이스들 간에는 서로 상호 운영 환경들이 상이하기까지 하다. 이와 같은 특징들로 인해 홈 네트워크 시스템은 데이터 호환, 동시 제어, 동적 플러그인과 같은 결정적인 기능들을 지원하기가 매우 어렵다. 따라서 홈 네트워크 시스템은 상대적으로 재사용성이 낮은 편이다. 본 논문에서는 홈 네트워크 시스템의 재사용성을 향상시키기 위해 가변성들을 지원할 수 있는 재구성 가능한 프레임워크 설계 기법을 제안한다. 이를 위해 홈 네트워크 시스템의 가변적인 부분들을 추출하여 이러한 가변부들을 가변성 타입들로 정의한다. 그리고 이러한 가변부들을 반영한 재구성 가능한 프레임워크의 구조를 설계하고 선택 기법과 플러그-인 기법을 통해 재구성 가능한 프레임워크를 특화시킬 수 있는 기법을 제시한다. 또한 제안된 프레임워크를 실제 홈 네트워크 시스템에 적용함으로써 재사용성과 제안된 설계 기법을 평가하고, 재사용성 평가 메트릭을 이용해서 이러한 사례 연구들의 결과를 평가하여 분석하고자 한다. 본 연구의 제안된 기법이 평가 결과를 분석한 결과 현존 시스템보다 재사용성을 보다 향상시킬 수 있음을 기대한다.

Key Words : Framework, Configurable Framework, Variation Point, Customization, Reusability, Variability

*교신저자 : 조은숙(escho@seoil.ac.kr)

접수일 11년 02월 24일 수정일 11년 03월 15일 게재확정일 11년 04월 07일

1. Introduction

Embedded software, implemented on the microprocessors, is the core software which is to diversify the functions of industrial and military control devices, digital customer electronic devices, and automatic sensor devices, and to increase value-added. Embedded operating system, embedded middleware, embedded applicable software, and embedded software development tool are included in the area of embedded software. Among the embedded system, the most typical example can be home network systems. Home network systems refer to the systems which exchange each of their data through the network connection of the devices, such as customer electronic devices, personal computers, and communication devices.

In the embedded software area, optimization technology is considered the core technology that influences the product value. For instance, the hardware of communication router costs hundreds of dollars, but its final price jumps to thousands of dollars once all sorts of communication protocols and control software are embedded on the hardware. In the future, while the computing environment develops into ubiquitous one, the importance of embedded software will be highlighted[1,2].

However, practicable and systematic study about embedded software is yet in the early stage. That is, in developing embedded software, software engineering techniques are not fully reflected. For instance, there are few cases systematic construction processes or techniques applied in software development, and various techniques for embedded software design are not brought up. In particular, in the development of home network systems software engineers need to consider the variability of various devices or protocols that will be used for home network systems to increase the reusability or efficiency of their software. However, current home network systems have been designed and implemented with little consideration of the variability described above[3-5].

Understanding this issue, we will suggest the configurable framework that can efficiently support the variability area through the static and dynamic meta-model. In case of developing embedded software, proposed framework can change dynamically various devices, one of variability characteristics of home network software. Consequently, it not only enhances the reusability of the embedded system, but also reduces the system complexity. This paper consists of the following: Section 2 focuses on the limitations of the home network system development and issues when applying the design technique of the existing variability into home network systems; Section 3 suggests the categorization of the variation point of the home network system which the paper suggests, the configurable framework design technique; Section 4 suggests improving reusability techniques based on configurable framework Section 5 explains case studies applying the proposed reusability framework and proposes the experimental results and evaluation metric. We then conclude the paper with key observations in Section 6.

2. Develop and Design Issues in Home Network System

In existing home network systems, the client/server model is chosen to connect their each device with servers through network settings. Since this approach is required to change the network settings for each device when the network environment of each device is changed, it may not support the portability of the home network system. Besides, in IPv4 address systems, it is not easy to allocate new IP addresses to newly registered devices to a home network. Also, customer electronic devices possess their own control data formats chosen by each of their manufacturing company. Thus, those devices with different data formats cannot coexist within one network in existing home network systems.

One of the biggest problems of home network systems is the interoperability among the customer electronic devices. That is, interoperability among digital devices must be guaranteed. Home network middleware systems are developed for addressing the interoperability issue, but still remains at its early stage.

The second problem is that the home network system lacks of the mechanism to control the various customer electronic devices at the same time. That is, the simultaneous control mechanism that can control the different devices from an event at the same time should be provided. Especially, this part must be managed by software and dynamic customization techniques that can

dynamically control the devices need to be considered.

The third problem is that the home network system is not considered commonalities and variabilities in protocols, device types, and services. In order to efficiently support these parts, commonality areas should be designed and developed as reusable unit components, while variability areas should be designed to be dynamically plugged in. However, since most solutions are designed and developed for each product, the reusability rate for the same devices or services remains very low. This paper provides a configurable framework and reusability improvement techniques to increase the reusability of home network systems.

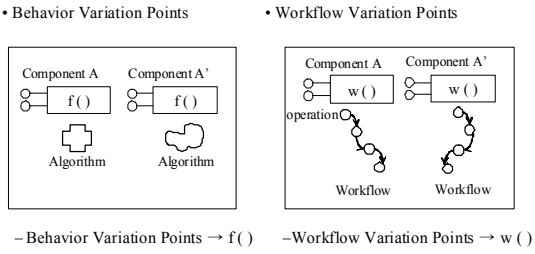
Reusable component must provide variation point to correspond to the required matters of various domain assignments. This variation point can improve not only component's reusability, but also can apply components into various domains[6,7]. On the other hand, mechanisms and design patterns for the existing variation point is limited and inadequate in satisfying the various required specification of the fast-changing system. This paper suggests a configurable framework which can fulfill the various requirements of home network embedded systems. This configurable framework can reflect on the designed parts as variation point and correspond to the various requirements not only for development phase but also in operating. The proposed configurable framework of home network embedded systems can increase its reusability and reduce development time and cost.

3. Configurable Framework Design based on Static Meta-Model

This proposed framework is intended to improve the reusability of home network embedded systems and can satisfy the various requirements during the development of various home network systems[8]. Also, light weight-home network systems can be developed and can dynamically be changed not only during their development phase but also during their operation. The proposed technique can provide the variability in embedded software areas to support hardware's changement of home network systems.

3.1 Variation Point of Home Network System

When developing the projects of various domains, the areas where the changes to certain requirements frequently occur are often called as variation points. In Fig.1, this variation point can be defined as behavior and workflow variation points. Also, like Fig. 1, the behavioral variation point provides the identical operation names but different algorithms for the f() function of two components that provide the similar function. As the above, the variation point provides different functions for certain requirements. Workflow variation point is the case in which the identical operation w() provides different function call flow.



[Fig. 1] Behavior and Workflow Variation Point

In home network systems, customer electronic appliances have control data format according to each choice made by their manufacturing companies. There are diverse types and protocols of digital device. Also, operating environment among the devices is different. In this paper, we extract these various parts as variation point of home network system and classify them as the kinds of variation point like the following. The kinds of variation points of home network embedded systems are Control, Device driver, Device protocol, AV Codec, and Operating systems (OS) is depicted. In particular, control variation point is the kind to manage the identical behavior differently by device. Device driver variation point and device protocol variation point are those defined to control the devices composing of home network system and the specialized protocol according to them. AV Codec variation point is the variation point to convert data format into the appropriate forms for each device, due to diversified data format control. Operating systems

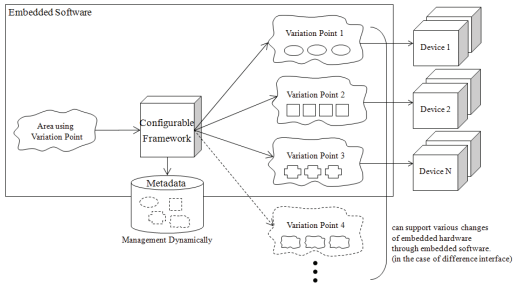
variation point controls operating mechanism for mutual linkage among various devices within home network systems. This paper defines not only variation points itself, but also the areas using these variation points.

Those areas using variation points must follow the certain conditions (e.g. Variation point identifier) to utilize the diversity of variation points.

3.2 Configurable Framework Design

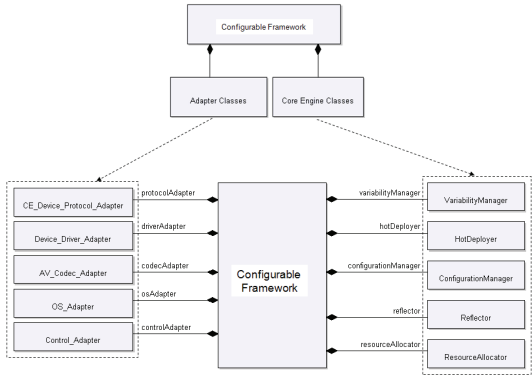
In this section, we design a configurable framework based on static meta-model using UML's class diagram or component diagram[9-11]. In Fig. 2, framework to improve the reusability of embedded system allows variation point using area to use variation point through configurable framework. In addition, while the existing technology provides reusability through a single interface, configurable framework provides the base that can provide various interfaces. In this paper configurable framework means reusability framework in terms of improving reusability.

In Fig. 2 above, this configurable framework intermediates between variation points and the areas using them. Variation point using area does not call out variation point directly, but its service through configurable framework.



[Fig. 2] Variation Point Management Mechanism followed by Configurable Framework

Variation point controlling mechanism of home network embedded system approaches variation point as variation point using class does through configurable framework. Variation point controlling mechanism also provides mechanism that can variably access variation point of home network system including device protocol, device driver, AV Codec, OS, and Control(Signal). Composition of configurable framework for home network embedded system is shown in Fig. 3 : it is composed of the adaptor relaying the variation point of home network system and the core engine classes for supporting variation point control.



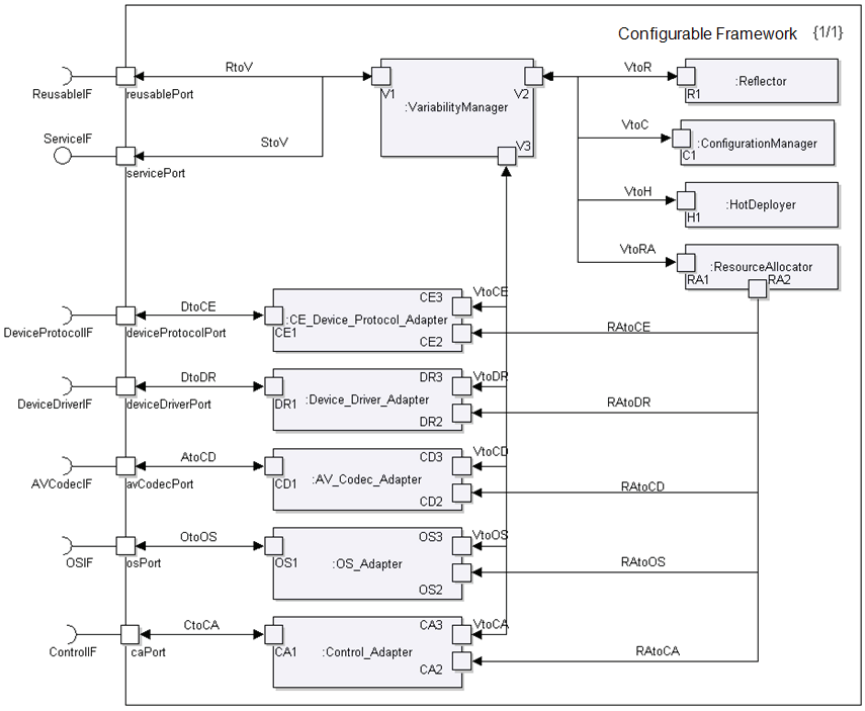
[Fig. 3] Composition of Configurable Framework

Variation point controlling mechanism of home network embedded system approaches variation point as variation point using class does through configurable framework. Variation point controlling mechanism also provides mechanism that can variably access variation point of home network system including device protocol, device driver, AV Codec, OS, and Control(Signal). Composition of configurable framework for home network embedded system is shown in Fig. 3 : it is composed of the adaptor relaying the variation point of home network system and the core engine classes for supporting variation point control.

Adaptor of configurable framework performs as an agent about variation point of home network system: device protocol, device driver, AV Codec, OS, control variation point. The core engine classes to control variation point are variability manager, reflector, configuration manager, hot deployer, and resource allocator.

A configurable framework on variation points of home network systems can dynamically provide home network services through adaptors and core engine classes.

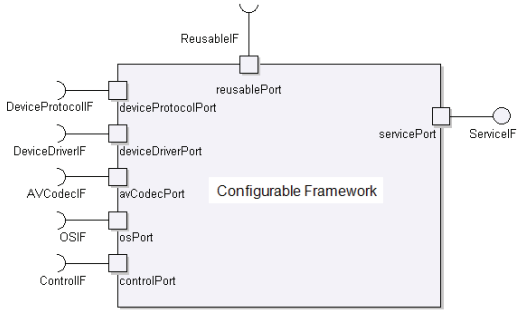
The adaptors of the configurable framework and internal structure of the core engine classes are shown in Fig. 4. Adaptors on the variability of home network



[Fig. 4] Internal Structure of Configurable Framework

systems decide which functions to use through ReusableIF and the variability manager of the configurable framework performs as an agent about configuration and service calls. Configurations by the variability manager are accomplished through the reflector, the configuration manager, and the hot deployer and the configured service is provided by the variability adaptors. The resources management of home network embedded systems are configured by the resource manager and differently by the variability adaptor. For example, the resource manager allocates different resources, such as memory depending on each operating systems(e.g. Window CE and VxWorks) that the operating systems variability adapter configures.

In order to provide change of variation point, this configurable framework provides the interfaces in illustrated Fig 5. ServiceIF provides services through required interfaces of the configurable framework. Required interfaces on variation points of home network systems can be configured by diversified variable functions and this configuration is fixed through ReusableIF.



[Fig. 5] Interface to manage Variation Point of Home Network Embedded System

The followings are the characteristics about internal constituent elements in configurable framework.

◆ Variability Manager

Variability manager intermediates and helps use variation point and acts as interface of configurable framework.

Variability manager calls variation point through variation point identifier. Since variability manager calls it

by identifier, the area using variation point does not affect to change of variability adapter or class inside of configurable framework.

◆ Configuration Manager

Configuration manager manages meta-information of variation point which will be used by variation point using area. Configuration manager calls variation point metadata through variability identifier of variation point, and this identifier defines in variation point using area and delivers to the configuration manager through variability manager. As [Fig. 6], configuration manager obtains information about the detailed variation point metadata-based on variation point identifier delivered in variation point using area.

```
Object result = VariabilityManager.execute(_VARIABILITY_NAME_, _PARAMETER_);
```

_VARIABILITY_NAME_ : Variation Point Identifier

[Fig. 6] Variation Point Manager Execution Code

Since configuration manager controls metadata with XML base, it can manage dynamically metadata and the configuration manager is the tool that can satisfy the requirements which have to change into diversified devices depending on the domains just like the traits of home network system.

◆ Metadata Repository

Metadata repository is the place that includes metadata of variation point and manages variation point information based on XML.

```
…

<Variability Identifier = "_VARIABILITY_IDENTIFIER_">
    <Used-By> _ADAPTER_NAME_ </Used-By>
</Variability>


<Adapter Name ="_ADAPTER_NAME_">
    <Class> _CLASS_NAME_ </Class>
    <Behavior> _BEHAVIOR_NAME_ </Behavior>
    <Context> _CONTEXT_INFORMATION_ </Context>
</Adapter>
…
```

[Fig. 7] Variation Point Metadata

In Fig 7, the metadata of the variation point consists of variation point identifier, variability adapter name, class name providing functions of variation point, name of behavior, and context information of action circumstances.

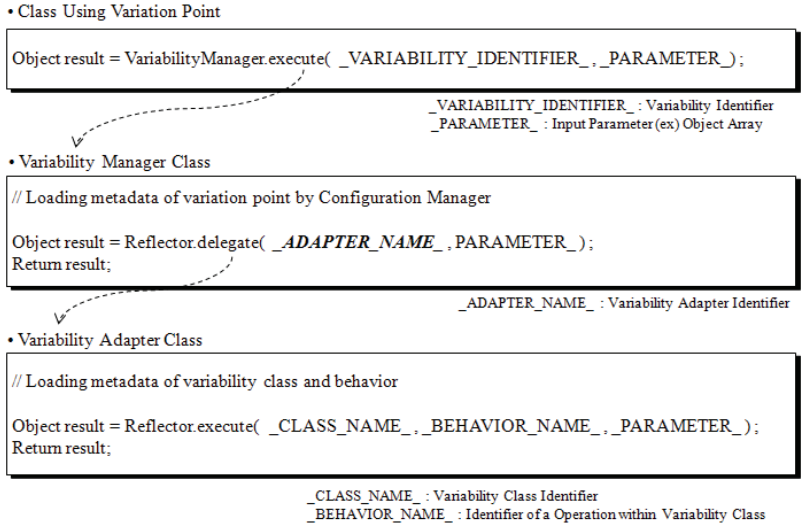
Variation point identifier is used as the identifier to call variation point in variation point using area and define in code of variation point using area. Variability adapter name, as a constituent of configurable framework, acts an intermediating role to dynamically change into the certain class among variation point class. Class name represents class that can be chosen through adapter and behavior name represents the behavior(operation, mathematical function) of the chosen class. This paper defines a meta-information as a metadata that can call multitude class and multitude behavior through multitude adapter. This feature of metadata provides the base that can dynamically change the various functions in the paper.

◆ Reflector

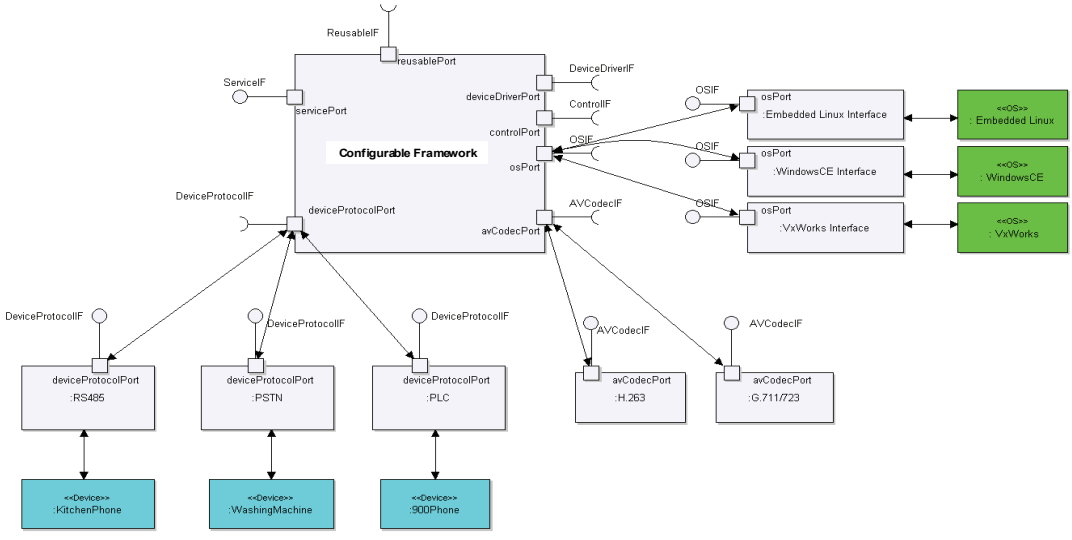
Reflector, as a tool to call physical variability adapters or variability classes through the metadata of the variation point, is based on the reflection function in order to dynamically call classes. Reflection is the mechanism that calls the function of physical classes by providing the meta form of the class name (String type) and the behavior name (String type), and input parameter (Object Array type).

This reflection mechanism is provided in standard development platforms (e.g. J2EE and .NET), and the reflector of this configurable framework customize this mechanism to provide functions that can manage metadata of variation point. In this manner, reflector of configurable framework not only helps dynamically change the various behaviors of variation point class but also various interfaces' class.

The reflector executes the physical variability adapter using the variability adapter identifier. The variability adapter also executes the variability class through the reflector. For the changes to variation points, only the meta-information of the variation point should be changed and variation point code in Fig. 8 is not influenced. The variability of most home network systems can provide variation points in this structure and the changes to device protocols or device driver classes are also possible



[Fig. 8] Variation Point Selection and Execution code



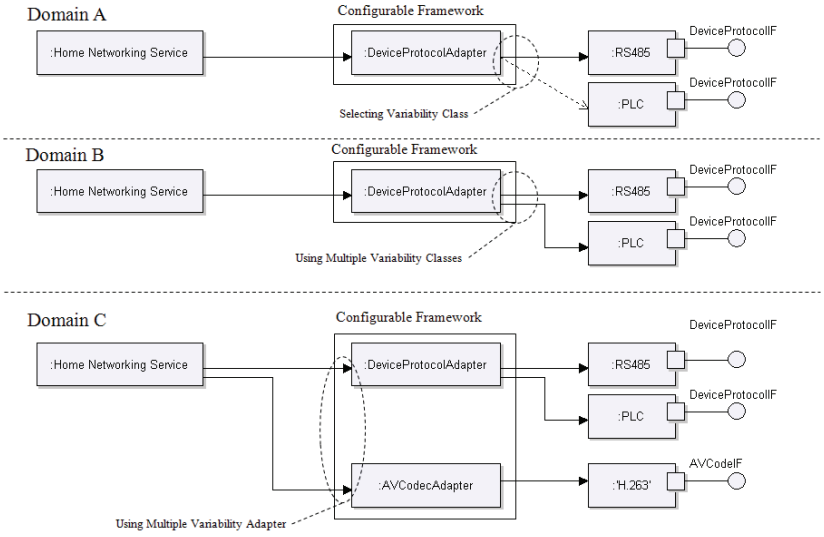
[Fig. 9] Variation Point Structure

through this structure.

◆ Variability Adapter

Variability adapter intermediates variation point and the selected variability adapter calls class defined in metadata.

Variability adapter of configurable framework is defined as required interface to support managing diversified variable and provides variation point through class suitable for it. Fig. 9 demonstrates variability of home network system ― device protocol, AV Codec, and the design structure of variation point about operating system variability. Since home network system supports the various devices through the various device protocols, it has to provide an opportunity to variably select device-fitting protocol such as RS485 or PSTN, PLC and so on. Classes providing this device protocol needs to be designed to satisfy DeviceProtocolIF, the variability



[Fig. 10] Variability Adapter Structure

required interface. AV Codec and operating system variability also designs required interface on this wise.

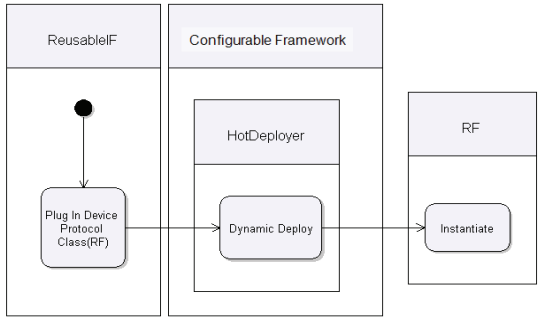
Once metadata of adapter changes, its physical class changes as well and provides different functions. Contrary to the existing studies, the paper states that application developers can dynamically change class adapter and call a very different function from that of class using variation point by changing adapter.

Like Fig. 10, according to domain requirements (Domain A, B), multiple adapter can select variation point class through DeviceProtocolAdapter or select multiple variation class at the same time and use its service. In case of home network system, when multiple adapter must deliver signals to the various customer electronics devices at the same time, this variable service structure is required. In case when requiring different forms of variability service, just like Domain C, multiple adapter must be able to use the multiple variability adapters at the same time. In home network system, multitude adapter can variably access AV Codec and use customer electronic devices. The difference from the existing study which this paper has is that multiple functions providing through the multiple adapter above is different characteristics from the existing study.

◆ Hot Deployer

Hot deployer is the tool to plug function class that is provided in system outside when inside of home network system cannot provide variability into the internal system. The concept of plug does not insert class within embedded system package but means that class instantiate compatible to the system operating environment.

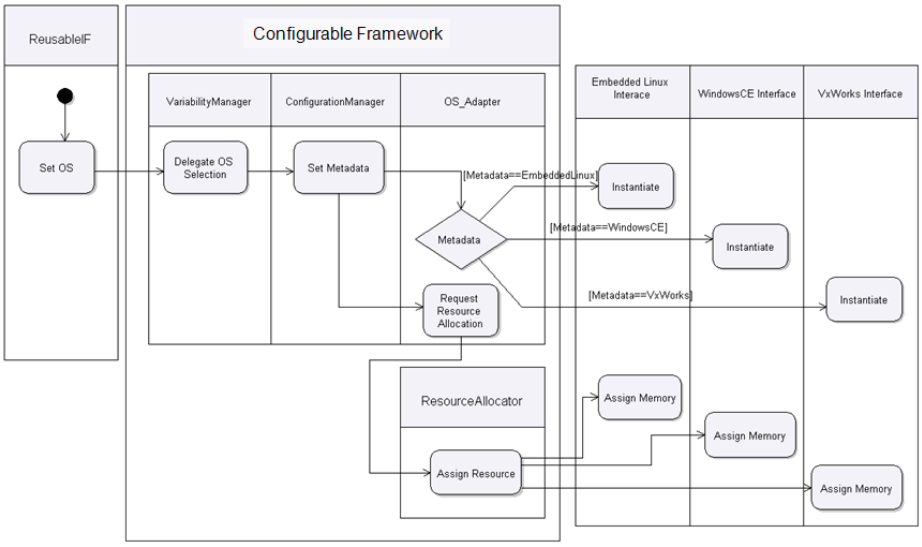
As shown in Fig. 11, in case when providing device protocol variability from the outside of the system through configurable interface, it needs to instantiate the object of variation point class (RF) as required by its dynamic deployer and make it possible to access variably from the existing operating system.



[Fig. 11] Variation point class plug-in by hot deployer

◆ Resource Allocator

As Resource Allocator is a tool to support resource allocation according to the change in operating



[Fig. 12] Variability Management Flow

environment of the embedded system, in case of home network systems, it mostly configures memory size or voltage followed by the change in operating systems.

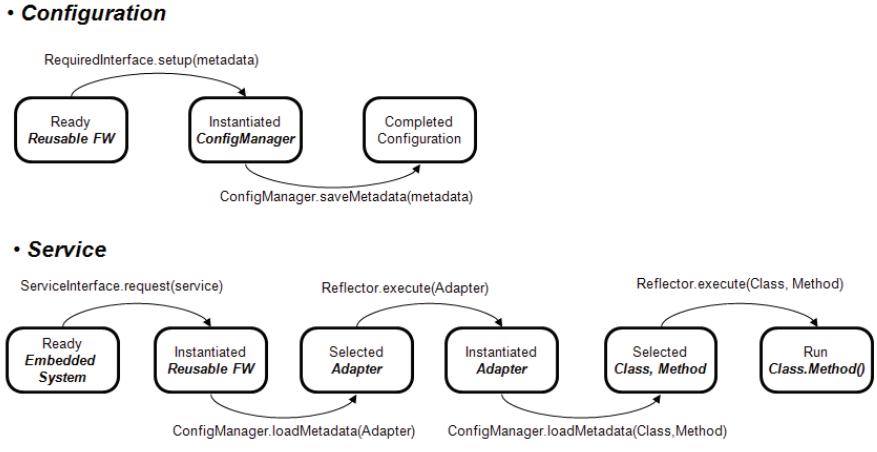
Fig. 12 shows that when operating environment of home network systems has to be changed to Embedded Linux, Windows CE, or VxWorks according to the domain requirements, the resource allocator configures the memory of embedded systems corresponding to the required operating systems. This configured resource information can be applied when the home network system is configured (or deployed) in the domains.

4. Reusability Improvement Technique

The paper suggests the two techniques to change variation point of home network embedded system: one technique is to select variation point through configurable framework and the other one is to plug in variation point required from the outside of embedded system.

4.1 Selection Technique

Variation point selection technique is the technique that changes the variation point by selecting one class among



[Fig. 13] Configuring Variation Point (Selection Technique)

variation classes within home network embedded software.

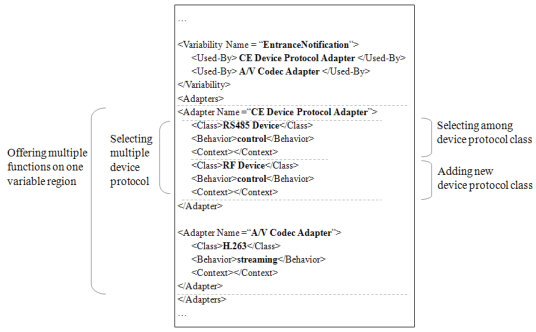
Selection technique is a technique that changes selection among variation points. Selection technique can design of configurable framework in configuration side and service side.

Variation point configuration by selection technique is depicted in Fig. 13, and Configure Manager of configurable framework is inputted meta-information from the required interface. This meta-information is information among function (device, protocol) that become design (or implementation) within embedded system.

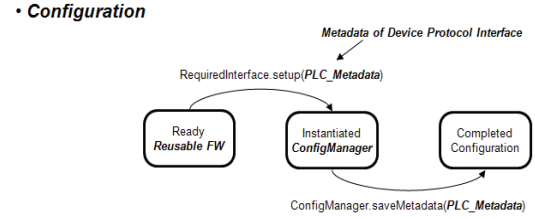
Meta information that is established depending of selection technique has structure of XML form with Fig. 14 and include establishment information to class, function from adaptor of variation point. This meta-information is called to service time, can offer function of established variation point.

Fig. 15 expresses configuring protocol of home network system.

Because home network system supports various device through various device protocol, should offer so that can select RS485, PSTN(Public Switched Telephone Network), PLC(Power Line Communication) and so on that is appropriate protocol to device as variable.

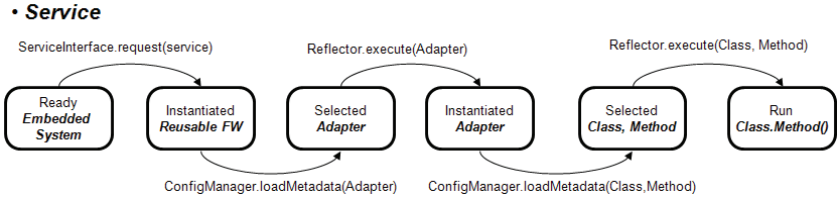


[Fig. 14] Meta Information

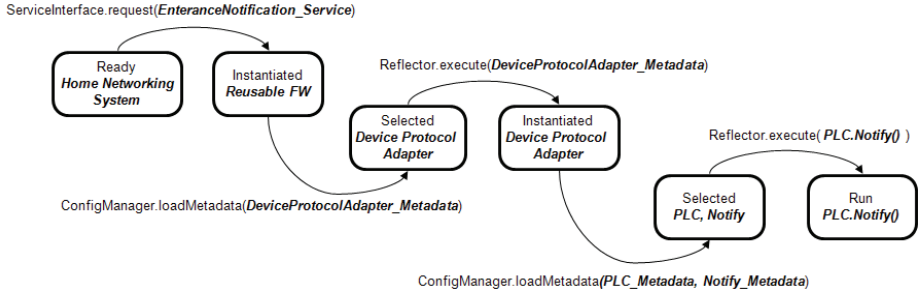


[Fig. 15] Configuring Protocol (Selection Technique)

Meta information that is established by selection technique can be serviced by configurable framework like Fig. 16. In case of request service had involved with variation point in embedded system outside, the service interface requests that offer by service that is established among variation point by configurable framework.



[Fig. 16] Providing Service of Variation Point(Selection Technique)



[Fig. 17] Providing Protocol Service(Selection Technique)

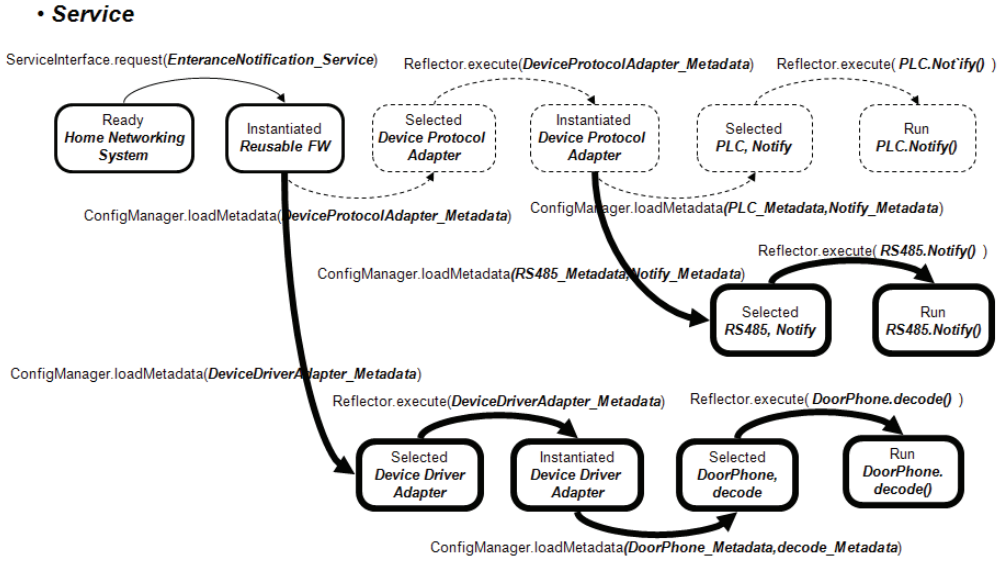
Configurable framework selects adaptor that delegate variation point using information that is established to meta-information and call class and function by selected adaptor.

Service for protocol variation point is same Fig. 17 in home network system. In case of request entrance visit service in entrance, home network system calls configurable framework because protocol service is related with variation point. Configurable framework selects protocol adaptor using meta-information that related with protocol. Protocol adaptor calls PLC service that is established among various protocols and provides entrance visit service.

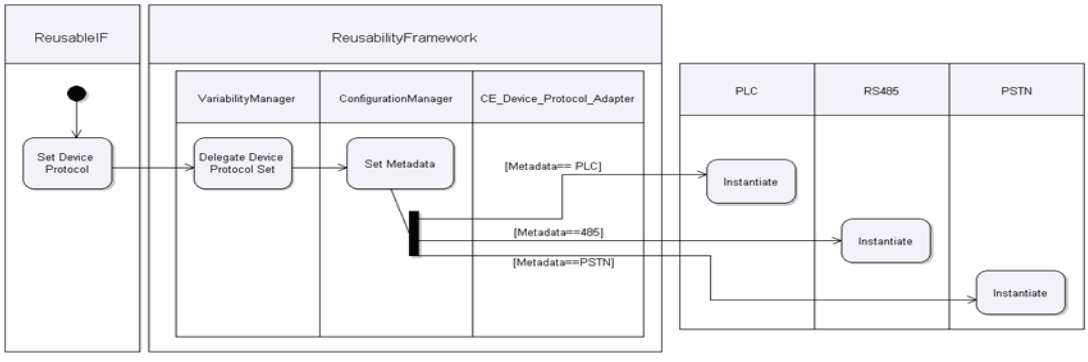
Composition service of configurable framework expresses for one service that various function can be changed.

As well as protocol alteration about exit and entrance visit service, can change driver for device alteration. These alterations expresses that amendment is available to entirely other structure. This study can accommodate various requirement because provide amendment in entirely other structure to differ that amendment is available in uniformity structure (or interface) within in existing study.

Fig. 19 demonstrates behavioral flow to provide variation point selection mechanism in home network embedded systems. In ReusableIF, once a device protocol is configured, the configurable framework configures the



[Fig. 18] Providing Composed Services(Selection Technique)



[Fig. 19] Behavior flow about device protocol variability

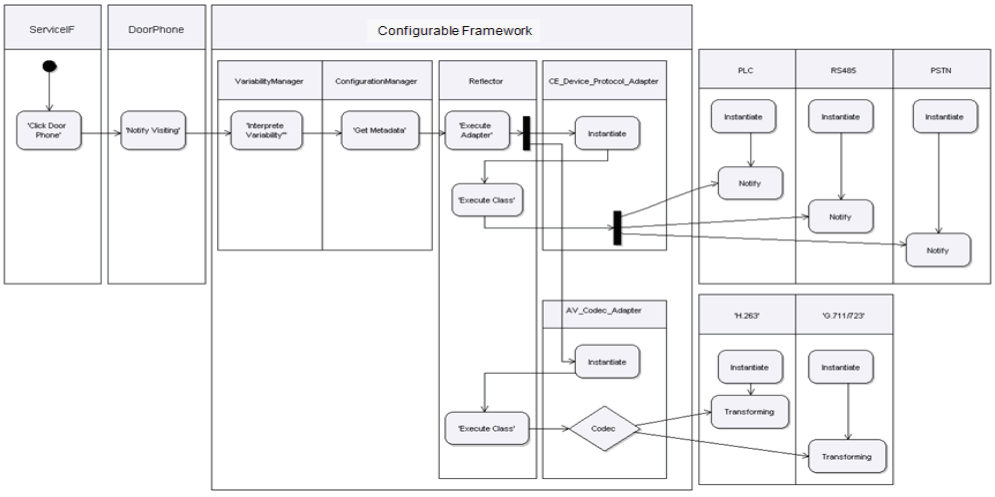
metadata about the device protocol, and instantiates the configured device protocol.

As shown in Fig. 19, the technique can configure various device protocols at the same time and home network system can provide various device protocols concurrently. As shown in Fig. 20, it can provide device protocol and variability on AV codec about certain service at the same time. In another specific domain, it can choose only one between two variabilities and provide service. Because of the attribute of home network system, the technique needs to provide various devices types and requires the technique to manage various variabilities at the same time since codec can be different by device.

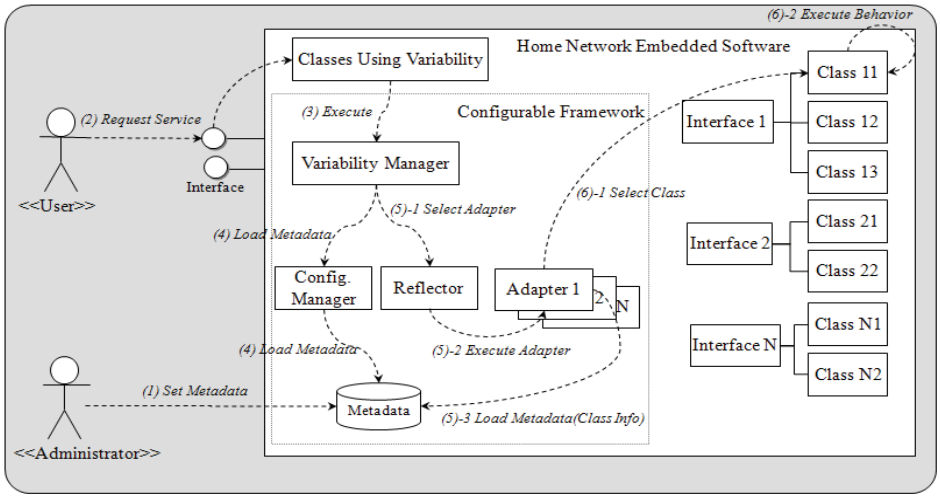
The detailed flow on configuration of variation point selection technique and its service is shown in Fig. 21 and Fig. 22, and we would like to define each step.

(1) Configurating Meta-information of Variation point

As illustrated in Fig. 21, the manager of embedded systems configures the metadata about variation points. The manager configures the information of interfaces and classes through the system specification about the specific variation point to be changed. The manager can also directly access the metadata repository in the form of



[Fig. 20] Variability Management Behavior Flow of Device Protocol and AV Codec



[Fig. 21] Variation Point Selection Technique

XML and configure such metadata as shown in Fig. 7.

(2) Requesting Service

The user requests the service of embedded systems. If the request service calls variation points, it calls the configurable framework just like Fig. 21. The class using variation points defines and calls its variation point identifier that can be used to obtain the metadata of the variation point.

(3) Calling Configurable Framework (Variability manager)

The areas using variation points call the variability manager of the configurable framework to call the variation point. The variability manager calls the metadata of the variation point based on the variation point identifier delivered by the class using variation point and also calls the variability adapter through the reflector.

(4) Calling Metadata Information

Variation point is called through variation point identifier in metadata repository. The corresponding metadata includes variability adapter connecting variation point, class and behavior performing the actual execution(operation), and context information. As shown in Fig. 7, metadata of variation point can call one class and several classes with a basis of variation point identifier ('_VARIABILITY_IDENTIFIER_'). Called metadata is delivered from variability manager to perform physical class.

(5) Selecting and Excecuting Variability adapter

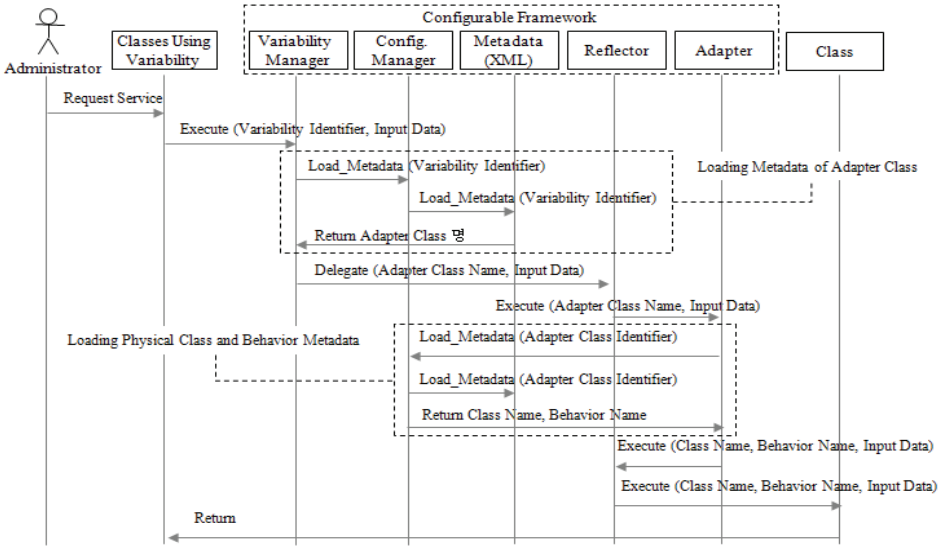
The variability adapter selects and executes the adapter that can be linked to the variation point based on the metadata called. Since the adapter access through the interface about the class of the variation point, it is not affected even though it changes into the other class. Like Fig. 23, the adapter calls the metadata of the variation point class to be executed.

(6) Selecting and Excecuting Variable classes

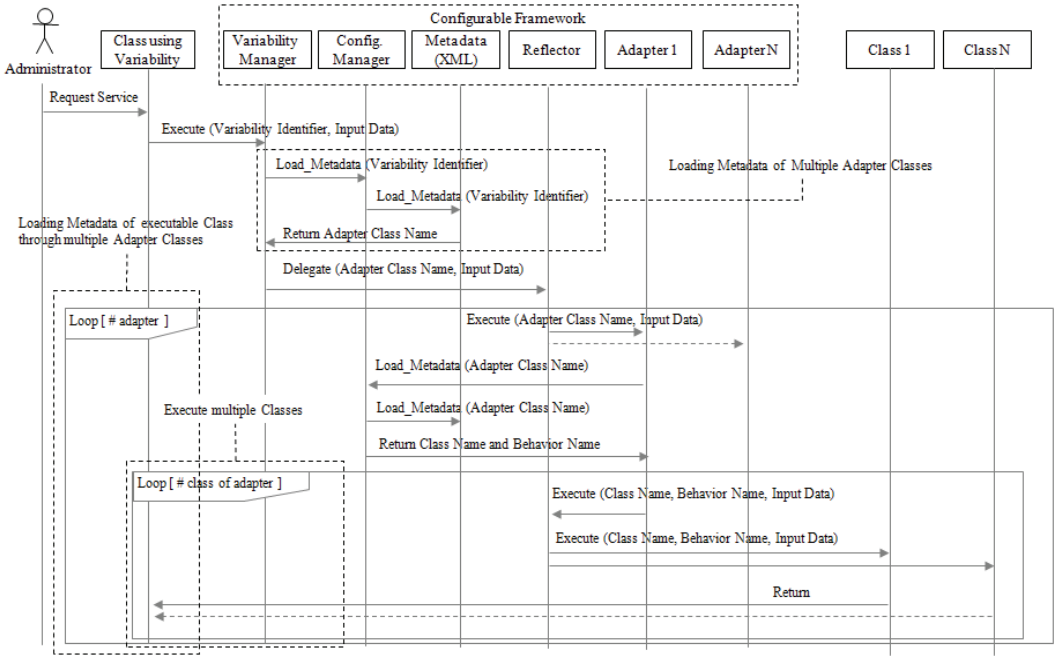
The adapter calls the behaviour (operation) of the variation point class chosen in the metadata. When the class behaviour is executed, input data is delivered from the class using the variation point, and output data is delivered to the class using the variation point through the configurable framework.

(7) Changing Variation point

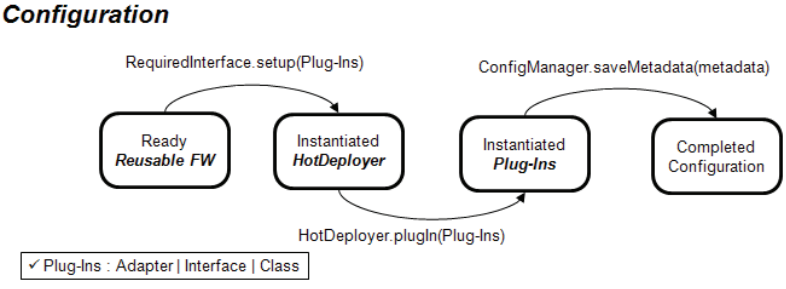
When changing variation point, change the variation point metadata of step (1) and perform from step (2) to step (7). Even if metadata in the step (1) changes, areas using variation point and variation point do not change at all. Only the configured metadata on variation point is changed.



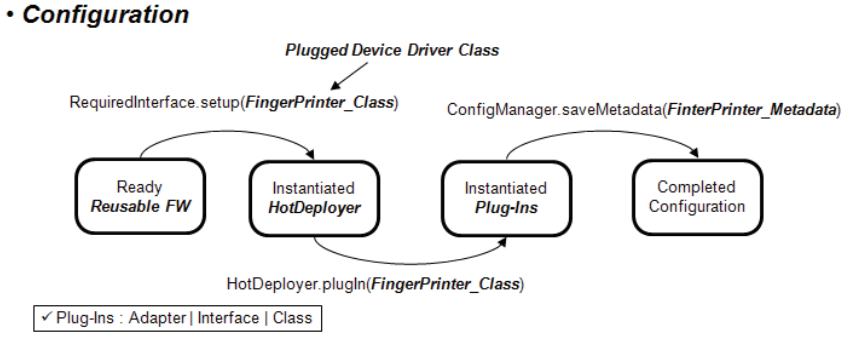
[Fig. 22] Sequence Diagram of Selection Technique (Single Class)



[Fig. 23] Sequence Diagram of Selection Technique (Multitude classes)



[Fig. 24] Configuring Variation Point (PlugIn Technique)



[Fig. 25] Configuring Device Driver(PlugIn Technique)

The differentiation point of configurable framework which the paper suggests is that it is not the limited change that changes implementation class on single interface, but changing into another form of interface and changing from the single to multitude interfaces is possible. Fig. 23 is the flow of variation point selection technique applicable to multitude class. Configurable framework can manage variability through a single variable adapter or do so by calling multiple adapters.

4.2 Plug-In Technique

Variation point plug-in technique is the technique that plugs in the variable class from the outside of the embedded system into its inside when the inside of the system cannot provide the variable requirements.

Plug-In technique is a technique that adds alteration function (device, protocol) into system inside in embedded system outside in case of can not offer alteration requirement in embedded system inside. Plug-In technique designs in configuration side and service side.

The configuration of variation point by plug-in technique input and configures function that try to change with Fig. 24. Input unit is Plug-Ins including the actual function that is not meta-information with adaptor, interface and class. Configurable framework makes Plug-Ins do activation (Instantiation, Activation) so that can provide service delivering hot deployer.

Home network system can change entrance function by existent key or door rock adding drive of fingerprint reconfiguer with Fig. 25 because various device can be connected about one function.

Service by plug-in technique is like Fig. 26.

In case of exit and entrance service is changed by fingerprint recognizer in door phone, Device driver adaptor requests entrance service to fingerprint recognizer.

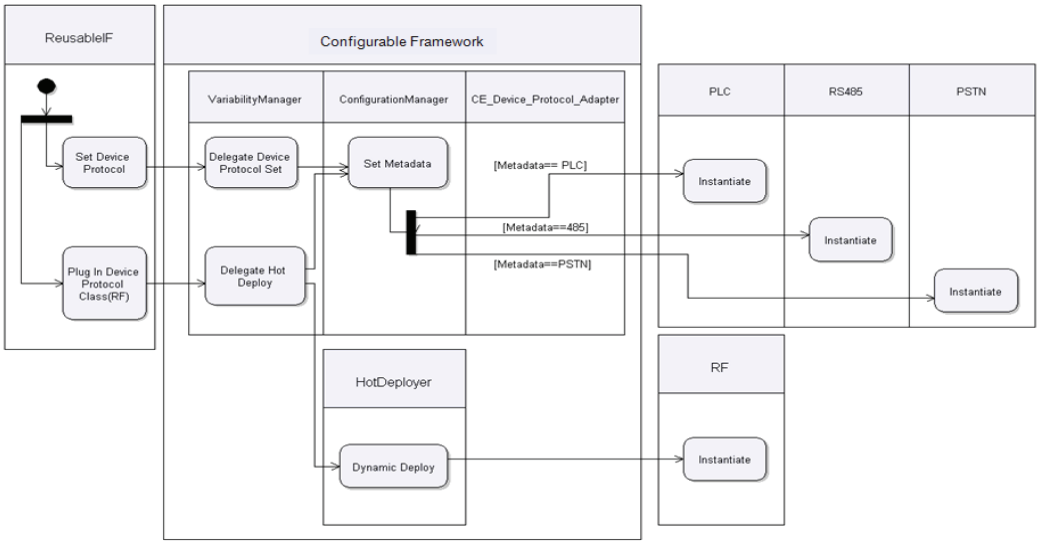
After new function (device, protocol) is a plug, dynamic metamodel about service is identical with selection technique.

The plug behaviour flow about device protocol variability is as the Fig. 27: if configuring device protocol(RF) in reusable interface(ReusableIF), configurable framework configures metadata about device protocol and hot deployer instantiates the configured device protocol.

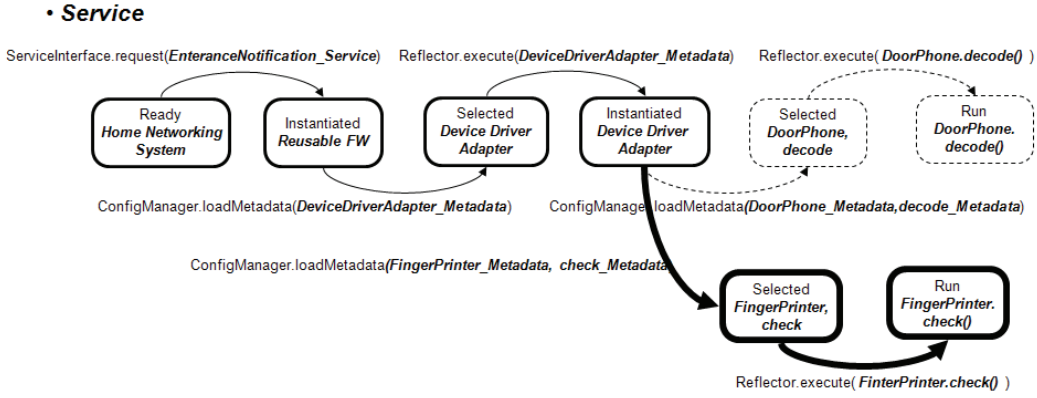
The detailed flow of configuration and service about variation point plug-in technique is shown in Fig. 28 and Fig. 29.

(1) Plug-In Variation Point Adapter and Class

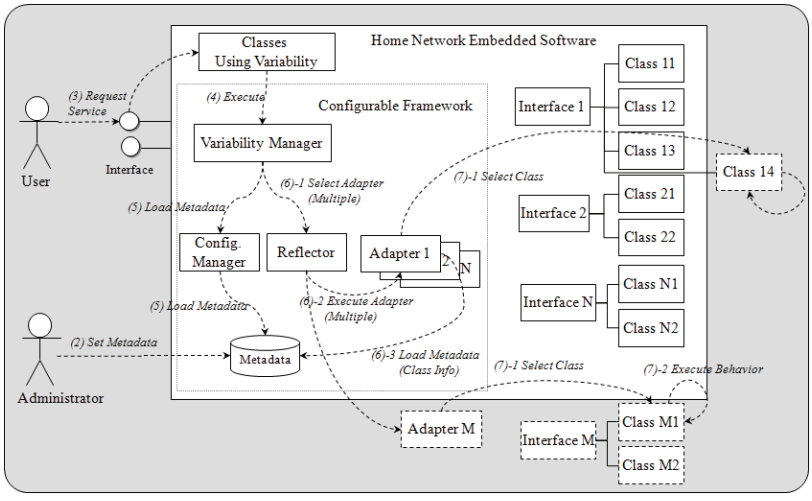
As shown in Fig. 29, configurable framework takes the new variability adapter and class by hot deployer taking object type as input parameter type. Inside of configurable framework, hot deployer activates the variability adapter and class which are dynamically plugged in.



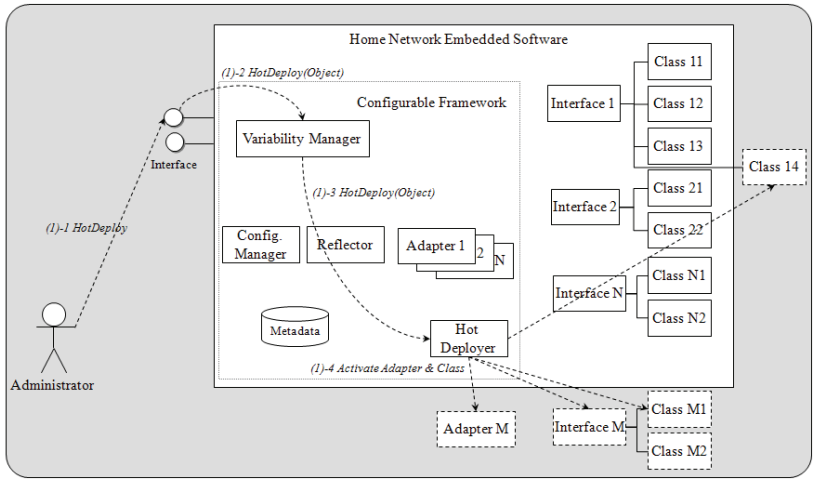
[Fig. 26] Plug-In Behavior Flow about Device Protocol Variability



[Fig. 27] Providing Service of Variation Points(PlugIn Technique)



[Fig. 28] Plug-In Technique



[Fig. 29] Variation Point Plug-In

(2) Configuration of Variation Point Metadata

Configuring metadata is identical with the variation point selection technique and in case of plug-in, it configures new class metadata since it is not defined in the breakdown of variation point which is to change.

(3) Service Request

The user requires the service of the embedded system. If the required service includes the variation point, it calls the configurable framework as shown in Fig 28. Variation point identifier defined in class using variation point of the embedded software does not consider whether the variation point is inside class (Hardware Driver) of the embedded software or not.

(4) Calling Configurable Framework (Variability Manager)

Class using variation point calls the variability manager of the configurable framework to call the variation point. The variability manager calls the metadata of the variation point based on the variation point identifier and calls the variability adapter through the reflector. The adapter called at this point may be either the variability adapter for home network embedded system within the configurable framework or the external adapter.

(5) Calling Metadata Information

Configuration Manager calls metadata in metadata repository through variation point identifier. Variation point identifier is not changed until it is used in class using variation point despite the change in variation point. The called metadata, which is the adapter, class, behavior(operation), context information showing variation point, is identical with the information of selection technique, but is the metadata for the external class of embedded system. Variation point metadata for plug-in technique calls the single and multiple classes through variation point identifier and embedded system can be constructed by composition of the inside classes and external classes.

(6) Selecting and Executing Variability Adapter

The variability manager selects and executes the adapter that connects the variation point based on the called metadata. The adapter accesses the interface on the class of the variation point, so the other areas using the variation point are not affected even if variation point is changed into another class. In case of plug-in technique, if metadata is configured to external adapter it is changed into another class type (Interface).

(7) Selecting and Executing Variable Classes

The variability adapter calls the behaviour of internal and external classes of the embedded system selected by metadata. When calling multiple classes, it cancall the class following the internal interface of embedded system or the external class by external adapter. Input and results of data is delivered through configurable framework.

5. Experiments and Evaluations

By using reusability evaluation metrics[12] to verify the reusability of the proposed framework, we will prove that reusability is improved by comparing the design using the proposed framework with that using the existing object-oriented method.

5.1 Evaluation Metrics

In order to evaluate the outcome of the case studies and the suggested framework, the paper defines as an extended metrics based on evaluation metric provided by Hironori Washizaki[13,14] and measured it by reflecting it to the case studies.

Definition 1. Understandability (EMI: Existence of Metadata)

EMI(c) means whether metadata exists or not in one component

$$EMI(c) = \begin{cases} 1 & \text{(Meta Info exists)} \\ 0 & \text{(otherwise)} \end{cases}$$

Confidence Interval : [0.5, 1.0]

Definition 2. Adaptability (RCC: Rate of Component Adaptability)

RCC(c) means the proportion of the customizable attributes tothe attributes in one component.

$$RCC(c) = \begin{cases} \frac{P_w(c)}{A(c)} & (A(c) > 0) \\ 0 & (\text{otherwise}) \end{cases}$$

where,
$P_w(c)$: number of writable properties in Component c
$A(c)$: number of fields in Component c
Confidence Interval : [0.17, 0.34]

Definition 3. Portability (SCCr: Rate of Component Portability)

SCCr(c) means the proportion of the business methods without return value to the business methods in one component.

$$SCC_r(c) = \begin{cases} \frac{B_v(c)}{B(c)} & (B(c) > 0) \\ 1 & (\text{otherwise}) \end{cases}$$

where,
$B_v(c)$: number of business methods without return value in Component c
$B(c)$: number of business methods in Component c
Confidence Interval : [0.61, 1.0]

Definition 4. Changeability

Changeability(c) means the proportion of the customizable methods to those of Required Interface in one component

$$Changeability(c) = \begin{cases} \frac{R_c(c)}{R(c)} & (R(c) > 0) \\ 0 & (\text{otherwise}) \end{cases}$$

where,
$R_c(c)$: number of changeable methods of Component c's required interface
$R(c)$: number of methods in Component c's required interface
Confidence Interval : [0.17, 0.34]

Definition 5. Replaceability

Replaceability(c) means the proportion of the customizable methods to those of required interface in one component.

$$Replaceability(c) = \begin{cases} \frac{R_r(c)}{R(c)} & (R(c) > 0) \\ 0 & (\text{otherwise}) \end{cases}$$

where,
$R_r(c)$: number of replaceable methods of Component c's required interface
$R(c)$: number of methods in Component c's required interface
Confidence Interval : [0.17, 0.34]

Definition 6. Extensibility

Extensibility(c) means the proportion of extendable methods to the methods of Required Interface in one component.

$$Extensibility(c) = \begin{cases} \frac{R_e(c)}{R(c)} & (R(c) > 0) \\ 0 & (\text{otherwise}) \end{cases}$$

where,
$R_e(c)$: number of extensible methods of Component c's required interface
$R(c)$: number of methods in Component c's required interface
Confidence Interval : [0.17, 0.34]

5.2 Experiments

The case study presented in this experiment shows the function that gives notices to the various customer electronic devices(Wall Pad, Kitchen phone, Set-top, Cell phone, etc.) at home when the visitor comes and door phone at the porch rings.

Because these customer electronic devices can be applied into domain differently, variant processing is needed. Therefore, this experiment proves that embedded system's reusability can be improved through variant processing of home network system.

The case study suggests the outcome estimating understandability, adaptability,changeability,replaceability, extensibility, and portability of metric through the existing object-oriented design method and the design case using configurable framework as suggested in the paper. Fig. 30 is design case for representing the arrival of the visitor on the kitchen phone through the relationship between

[Table 1] Design Case 1's Reusability Estimated Value

Factor	Measured Value
Meta Info	No
Number of Field	5
Number of Writable Properties	3
Number of Business Method	5
Number of Business Method without return value	4
Number of Method of Required Interface	0
Number of Changeable Method of Required Interface	0
Number of Replaceable Method of Required Interface	0
Number of Extensible Method of Required Interface	0

Metric	Measured Value	V_{Metric}	Measured Value
EMI	0	V_{EMI}	0
RCC	3/5=0.6	V_{RCC}	0
SCC_r	4/5=0.8	V_{SCCr}	1
Changeability	0/0=0	$V_{changeability}$	0
Replaceability	0/0=0	$V_{replaceability}$	0
Extensibility	0/0=0	$V_{Extensibility}$	0

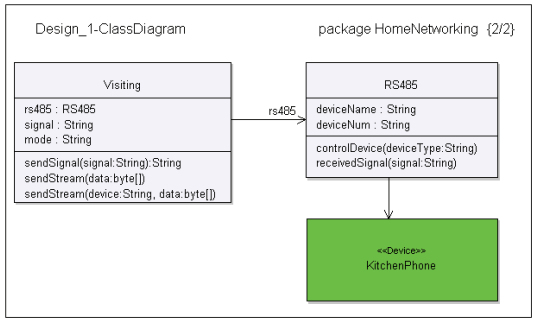
$$COR = 1.76 \times \frac{0+0+1+0+0+0}{6} - 1.13 = -0.84$$

class'Visiting'and class'RS485', the customer electronic device control protocol. The estimated outcome of reusability metric about this case study is as Table 1. Since metadata for providing understandability does not exist, EMI is 0, and because it is not in anywhere between the confidence interval [0.5, 1.0], VEMI is 0. ince there are three customizable attributes data among all the attribute data to provide adaptability, RCC is 0.6, and because it is not in anywhere between the confidence interval [0.17, 0.34], VRCC is 0. Since there are four business functions without return value among all the business functions to provide portability, SCCr is 0.8, and because it is in the confidence interval [0.61, 1.0], VSCCr is 1. Since there is no function to change among the functions of required interface to provide changeability, changeability is 0, and because it is not in the confidence interval [0.17, 0.34], VChangeability is 0. Since there is no function to replace among the functions required interface to provide replaceability, replaceability is 0, and because it is not in the confidence interval [0.17, 0.34], VReplaceability is 0. Since there is no function to extend among the functionsof required interface to provide extensibility, extensibility is 0, and because it is not in the confidence interval [0.17, 0.34], VExtensibility is 0. Understandability, adaptability, changeability, replaceability, extensibility, and portability in explained above are based on component reusability metrics and they are applied based on metric COR(Component Overall Reusability), thatis extended based on evaluation metric, suggested by Hironori Washizaki.

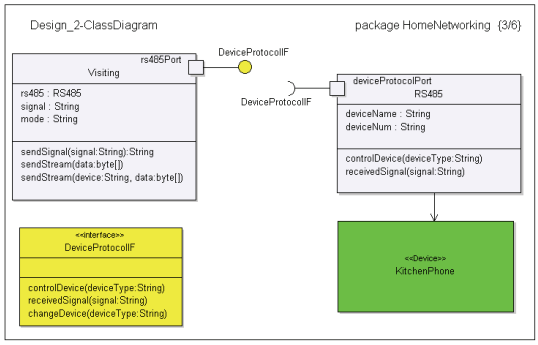
Reusability metric COR of Design Case 1 is -0.84. In generally viewed design, class 'Visiting' and class 'RS485' is strongly combined and it can be estimated that variability is too weak to control and reusability is low.

In entrance notification design case 2, as shown in Fig. 31, class 'Visiting' and class 'RS485,'the customer electronic control protocol, is not directly linked but through interface. The estimated outcome of reusability metric on this case is shown in Table 2.

Since the design examples in this case study arebased on all the identical data, their understandability, adaptability, and portability have the identical estimated outcome as shown in Fig. 30.



[Fig. 30] Entrance Notification Design Case 1



[Fig. 31] Entrance Notification Design Case 2

[Table 2] Design Case 2's Reusability Estimated Value

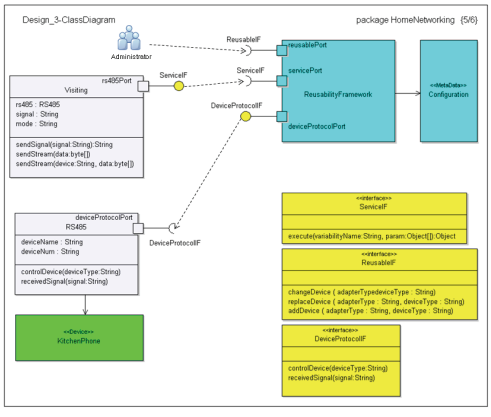
Factor	Measured Value
Meta Info	No
Number of Field	5
Number of Writable Properties	3
Number of Business Method	5
Number of Business Method without return value	4
Number of Method of Required Interface	3
Number of Changeable Method of Required Interface	1
Number of Replaceable Method of Required Interface	0
Number of Extensible Method of Required Interface	0

Metric	Measured Value	V_{Metric}	Measured Value
EMI	0	V_{EMI}	0
RCC	3/5=0.6	V_{RCC}	0
SCC_r	4/5=0.8	V_{SCCr}	1
Changeability	1/3=0.3	$V_{changeability}$	1
Replaceability	0/3=0	$V_{replaceability}$	0
Extensibility	0/3=0	$V_{Extensibility}$	0

$$COR = 1.76 \times \frac{0+0+1+1+0+0}{6} - 1.13 = -0.54$$

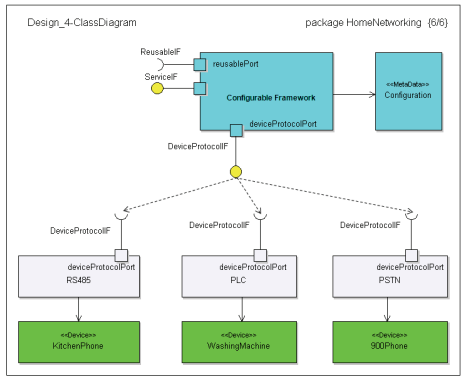
Since function to change is only one among the functions of required interface, changeability is 0.3, and because it is in the confidence interval[0.17, 0.34], VChangeability is 1. Since function to replace is none among the functionsof required interface, replaceability is 0, and because it isnot anywhere in the confidence interval[0.17, 0.34], VReplaceability is 0. Since function to extend is none among the functions of required interface, extensibility is 0, and because it isnot anywhere in the confidence interval[0.17, 0.34], VExtensibility is 0.

As the above, based on understandability, adaptability, portability, changeability, replaceability, and extensibility, reusability metric COR is -0.54. Since this design is the interface combined of class'Visiting'and class'RS485,'it has a higher reusability than the design case study 1. However, reusability metric COR is only reusable when it is larger than 0, it is not enough to be reused in general. The design case using 'Configurable Framework' as suggested in the paper is shown in [Fig. 32].

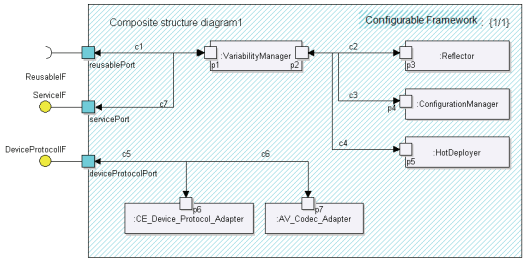


[Fig. 32] Entrance Notification Design Case

As shown in Fig. 33, it can support various device protocols through 'Configurable Framework'. Protocol class such as 'RS485', 'PLC', 'PSTN' follows 'DeviceProtocolIF'interface and the various protocols (PLC, PSTN, RF, etc.) can be variably managed in 'Configurable Framework.'



[Fig. 33] Design to support Various Device Protocols



[Fig. 34] Internal Structure of Configurable Framework (Composite Structure Diagram)

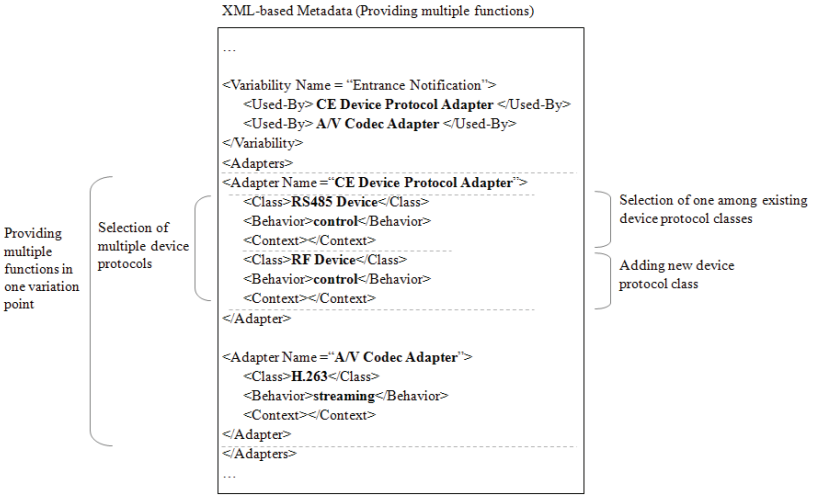
Fig. 34 is the Composite Structure Diagram which designed the internal structure of 'Configurable

[Table 3] Configurable Framework Design Case's Reusability Estimated Value

Factor	Measured Value
Meta Info	Exists
Number of Field	5
Number of Writable Properties	3
Number of Business Method	5
Number of Business Method without return value	4
Number of Method of Required Interface	3
Number of Changeable Method of Required Interface	1
Number of Replaceable Method of Required Interface	1
Number of Extensible Method of Required Interface	1

Metric	Measured Value	V_{Metric}	Measured Value
EMI	1	V_{EMI}	1
RCC	3/5=0.6	V_{RCC}	0
SCC_r	4/5=0.8	V_{SCCr}	1
Changeability	1/3=0.3	$V_{changeability}$	1
Replaceability	1/3=0.3	$V_{replaceability}$	1
Extensibility	1/3=0.3	$V_{Extensibility}$	1

$$COR = 1.76 \times \frac{1+0+1+1+1+1}{6} - 1.13 = 0.34$$



[Fig. 35] Device Protocol Variation Point Management Metadata

Framework'. As it provides the various adapters, various functions can be processed variantly.

The estimated outcome of the reusability metric, which applied the design case study using configurable framework, is shown in Table 3.

Since the design method has a basis on the identical data like design case study 1(Fig. 30) and design case study 2(Fig. 31), understandability, adaptability, and portability have the same estimated outcome.

Since function to change is only one among the functions of required interface to provide changeability, changeability is 0.3, and because it is anywhere in the confidence interval [0.17, 0.34], VChangeability is 1.

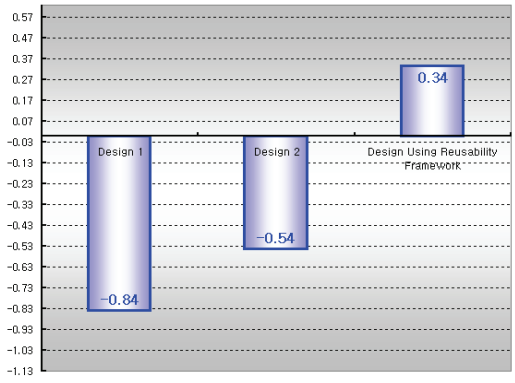
Since function to replace is only one among the functions of required interface to provide replaceability, replaceability is 0.3, and because it is anywhere in the confidence interval [0.17, 0.34], VReplaceability is 1.

Since function to extend is only one among the functions of required interface to provide extensibility, extensibility is 0.3, and because it is anywhere in the confidence interval [0.17, 0.34], VExtensibility is 1.

As the above, reusability metric COR based on understandability, adaptability, portability, changeability, replaceability, extensibility is 0.34. In general, the design shows that class 'Visiting'and class'RS485'are not directly connected but through 'Configurable Framework,' class 'Visiting' can provide various device protocols or other functions(various codecs) at the same time for entrance report.

Fig. 36 shows that the design case using 'Configurable Framework'can be said to have reusability because COR value is larger than 0[1], differently from design case 1 and 2 which are the existing design methods. From the aspect of structure, adding'Configurable Framework'may

increase complexity, but reusability still improves because changeability, replaceability, and extensibility increase compared to those of design case 1 and 2.



[Fig. 36] Reusability Estimated Value by Design Cases

6. Concluding Remarks

Home network systems should be able to support various devices to satisfy the requirements of the various domains. In this paper, we suggest a configurable framework in home network embedded software to support variable management of the home network system. Depending on device changes of the home network system, control, device protocol, device driver, operating systems, or AV Codec should be changed, and the proposed configurable framework provides interface mechanisms to dynamically manage these variable parts.

There are two ways to manage the variation points of the home network system: the selection technique selecting one among the variable functions within the home network system and the plug-in technique providing new functions from the outside of the home network system. These two techniques can be configured through meta-data and can support various requirements not only during the development phase but also operation time. This paper showed that the reusability of the home network embedded system improved by applying the proposed configurable framework and its reusability techniques to real-world home network embedded systems In the future research, we will work on the common configurable framework technique that can improve the reusability of the embedded system.

References

[1] H. Comma, "A Software Design Method for Real-Time Systems", Communications of ACM, Vol.27, No. 7, pp.938-949, Sept. 1984.

[2] J. Ready and D. Howard, "Structuring Real-Time Application Software Part1", VMEbus Systems, pp.33-45, April, 1991.

[3] Coplien J., Hoffman D., and Weiss D., "Commonality and Variability in Software Engineering", IEEE Software, pp. 37-45, November 1998.

[4] Weiss D. M., "Commonality Analysis: A Systematic Process for Defining Families," Second International Workshop on Development and Evolution of Software Architectures for Product Families, February 1998.

[5] Kang, K. C., Cohen, S. G., Novak, W. E. and Peterson, A. S., "Feature-oriented Domain Analysis (FODA) Feasibility Study", Technical Report CMU/SEI-90-TR-21, Software Engineering Institute (SEI), November 1990.

[6] Anastasopoulos M. and Gacek C., "Implementing Product Line Variabilties", Technical Report IESE Report No. 089.00/E, Version 1.0, Fraunhofer Institute for Experimental Software Engineering (IESE), Novermber 2000.

[7] Becker M, "Generic Components : A symbiosis of Paradigms", 2nd International Symposium on Generative and Component-Based Software Engineering (GCSE'00), Erfurt, October 2000.

[8] Szyperski C., Component Software: Beyond Object-Oriented Programming, Addison-Wesley, 2002.

[9] Rumbaugh J., et. al., Object-Oriented Modeling and Design, Prentice-Hall, 1991.

[10] Hopkins J., "Component Primer", Communication of the ACM Vol. 43, No.10 , October 2000.

[11] Atkinson C., Bayer J., Bunse C., Kamstices E., Laitenberger O., Laqua R., Muthig D., Paech B., Wust J., and Zettel J., Component-based Product Line Engineering with UML, Addison-Wesley, 2001.

[12] ISO/IEC JTC1/SC7 N2419 "DTR9126-2: Software Engineering - Product Quality Part 2 - External Metrics", 2001.

[13] Hironori Washizaki, Hirokazu Yamamoto and

Yamamoto and Yoshiaki Fukazawa, "A Metrics Suite for Measuring Reusability of Software Components", Proceedings of the Ninth International Software Metrics Symposium(METRICS'03), pp.1530-1435, IEEE, 2003.

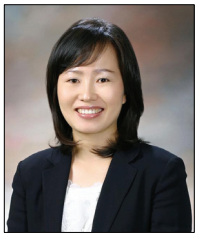
[14] Jeffrey S. P., "Measuring Software Reusability", IEEE Software, 1994.

Chul-Jin Kim [Regular member]

- Feb. 1996 : Kyonggi Univ., B.E.
- Feb. 1998 : Soongsil Univ., M.S
- Feb. 2004 : Soongsil Univ., Ph.D
- Sept. 2004 : Catholic Univ. Visiting Professor
- Dec. 2004 ~ Dec. 2009 : Samsung Electronics Co.
- Mar. 2009 ~ Current : Inha Technical College, Dept of Computer System, Assistant Professor

<Research Interests>
CBD, Component Customization, Embedded Software

Eun-Sook Cho [Regular member]



- Feb.. 1993 : Dongeui Univ. B.E
- Feb.. 1996 : Soongsil Univ., M.S
- Feb. 2000 : Soongsil Univ., Ph.D
- Jan. 2002 ~ Oct. 2003 : Invited Researcher in ETR
- Sept. 2000 ~ Feb. 2005 : Dongduk Women's Univ., Dept of Data Information, Full-time Instructor
- Mar. 2005 ~ Current : Seoil Univ., Dept of Software, Associate Professor

<Research Interests>
CBSE, Embedded Software, Service-Oriented Computing

Chee-Yang Song [Regular member]

- Feb.. 1985 : Hannam Univ. B.E
- Feb.. 1987 : Chungang Univ., M.S
- Aug. 2003 : Korea Univ., Ph.D
- May. 1990 ~ Sep. 2005 : Researcher in Center Research Institute of KT
- Oct. 2005 ~ Current : Kyungpook Univ., Dept of Software, Assistant Professor

<Research Interests>
CBSE, Model-Driven Architecture, Business Modeling, IPTV