Implementation Object Linking and Embedding for Processes Control Unified Architecture Specification on Secure Device

Yuankui Wang (Matr.-Nr.: 6670785)

University of Paderborn wangyk@mail.upb.de

Abstract. Object Linking and Embedding for Process Control Unified Architecture, known as OPC UA is the most recent released industry standard from OPC Foundation, which compared with his predecessors is equipped with a list of charming new features, with whose help OPC UA is capable of developing a common communication interface for devices which participate in automation system.Meanwhile, the technology of smart card is widely used in information security fields of finance, communication, personal and government identification, payment. Therefore it is meaningful and promising to develop OPC UA standard satisfying application on embedded smart card secure device, for the purpose of secure remote control, enterprise resource planning and etc. Since the storage and compute capacity of chip card is limited, OPC UA product will consist of two essential parts, namely client/server application code, realized as Android or other application, and communication stack, realized as Javacard Applet based on Remote Application Management from GlobalPlatform. The implemented demonstration scenarios and corresponding analysis show the possibility of developing OPC UA standard satisfying application on embedded devices with smart card to benefit customers.

1 Introduction and Motivation

According to the *Mobile Economy 2013* from *Global System for Mobile Communications Association*, at the end of year 2013 there are over 3.2 billion mobile subscribers in total, which means one half the population of the earth now enjoy the social and economic convenience brought by mobile technology. Moreover by year 2017 700 million new subscribers are expected to be added. And the number of mobile subscriber will reach 4 billion in 2018. Mobil technology opens nowadays a promising market.

Mobile products play an irreplaceable role at the heart of our daily life. With the help of mobile technology, the user's world in many domains such as, education, financial transactions, health and etc. are inter-connected. Mobile users are enjoying the advantages of mobility. Services, like 24/7 monitored home security, full control about the management of home humidity and temperature, exist not only in science fiction film but also could be realized by today's technology.

At the same time, mobility in industry and business world is also a critical assert, which can not only increase efficiency and productivity but also drive new revenue generation and competitive advantage. The most convicting example here is Machine to Machine communication, that is also referred as M2M technology. In M2M communication, machines which are usually embedded with smart cards exchange gathered date with each other to accomplish common task using wireless or wire networks. M2M technology is widely employed in different industry spheres such as factory automation, remote access control and sensor monitoring. It boosts the efficiency of corresponding processes, offers centralized service support and date management, minimizes system response time.

But in order to enjoy the aforementioned features, two tough issues must be resolved. First, how to achieve a common interface for the devices that participate in the system. And second how to guarantee system security under different communication environments with variant date complexity.

2 Solution Idea

In this master thesis, I am going to address solutions for questions mentioned in section *Introduction and Motivation* and design a smart home system for the purpose of demonstration. In this smart home system, home owner using smart phone is capable of experiencing 24/7 home security service, remotely managing inner home environment parameters and assigning access permissions. This system consists of smart phones with Universal Integrated Circuit Cards (UICC smart card), digital door locks, control devices, environment sensors and if necessary a central control computer. Moreover each device is equipped with smart card, which acts not only as secure token, that saves user credentials, but also is in charge of construction and management of the devices' communication.

In particular, I will introduce the newly released industry automation standards object linking and embedding for processes control unified architecture(OPC UA standards) to build a common communication interface for devices that are mentioned above and design communication stack for OPC UA standards on UICC smart card., whose duties are: creation and management communication between OPC client/server application, message serialization and secure message exchange.

In conclusion, OPC Unified Architecture is a platform independent industry policy, supports secure communication based on different network conditions between client and server that are provided by various vendors.

3 OPC Unified Architecture Structure Overview

3.1 OPC UA Specification

The whole OPC Unified Architecture specification can be divided into three main parts, core specification part, which consists of OPC UA concepts, security model, address space model, services, information model, service mapping

and profiles, access type specification part including date access, alarm and conditions, programs and historical access, at last utility specification part covering discovery together with aggregates.

In OPC Unified Architecture information that can be visited by clients is defined as address space[**?**] and there is a set of services[**?**] provided by OPC UA which are introduced in order to apply operations in the address space. The information in address space is organized as a set of in particular hierarchy structured objects. Clients can accept information provided by OPC Unified Architecture Servers in two major ways, binary structured data and XML documents, depending on the complexity of exchanged date, network quality and so on. In addition three kinds of transport protocol are already defined to support client server communication. They are: OPC UA TCP, HTTP/SOAP and HTTP. Also the hierarchy structure in which objects are organized in address space is also various according to OPC UA standards and not limited to simple single hierarchy.

Another charming feature of OPC UA is Event Notifications. With the help of Event Notification, OPC UA servers are allowed to immediately after some conditions are satisfied publish data, which is subscribed by clients. In this way, clients can for instance discovery failures within client-server-communication quickly and recover as soon as possible, which in return minimizes the lost to the smallest possible amount and also clients are able to observe the subscribed data more precisely and find the pink elephant as fast as possible.

3.2 OPC UA client/ server structure

Figure 1 illustrates a typical OPC UA client server architecture and also describes a combined server-client. The routine communication between client and server consists of requests from client, corresponding responses sent from server and notifications which are generated because of clients early subscription.

Figure 2 pictures one simple OPC UA client containing client application, an internal API, isolating the application code from communication stack, and a communication stack that converts API calls into messages and delivers them to OPC UA server.

In figure 3, one OPC UA Server structure is explained. As the aforementioned client structure, it also includes three main parts, server application, internal API and communication stack. It is worth mentioning that, real objects here are referred as physical field devices or software application that is only maintained internally. View, which is pictured as a part of address space, presents objects that can be browsed by clients

3.3 Secure Channel and Session

Since some data exchanged between client and server could be extreme precious and should be protected from other malicious third party, OPC UA defines a full set of security model, with which developer of system can configure the security

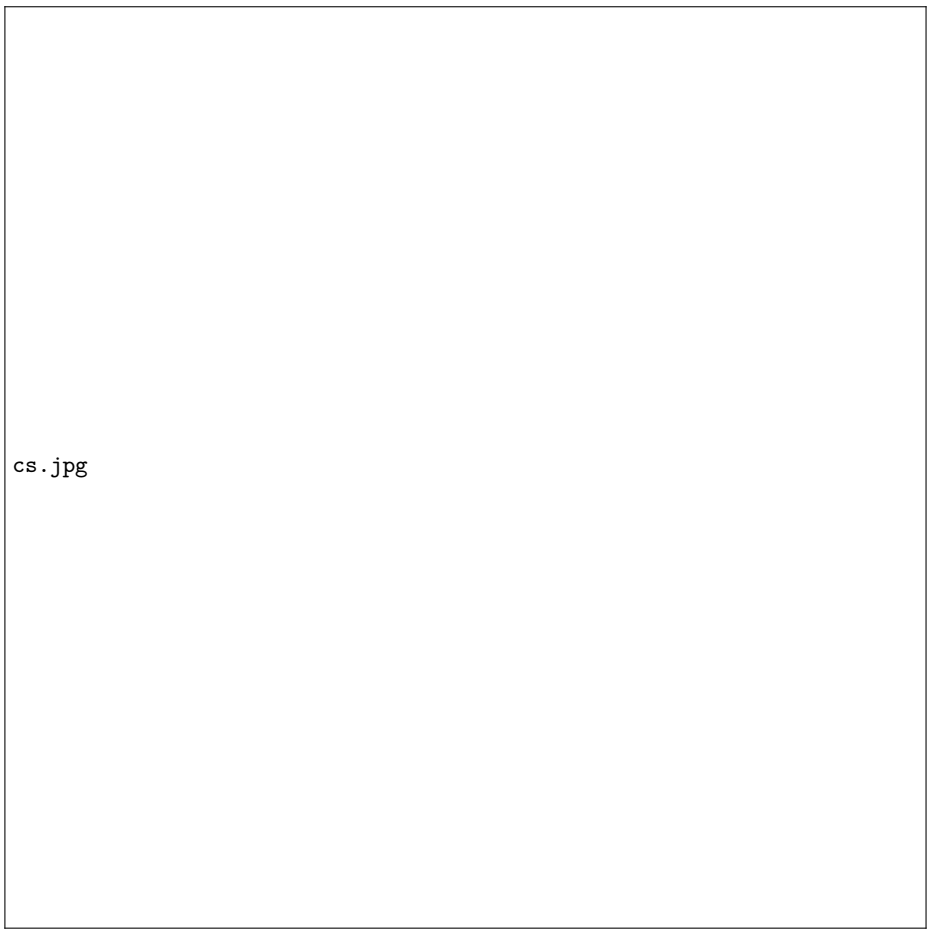

Fig. 1. OPC UA Client Server Structure[**?**]

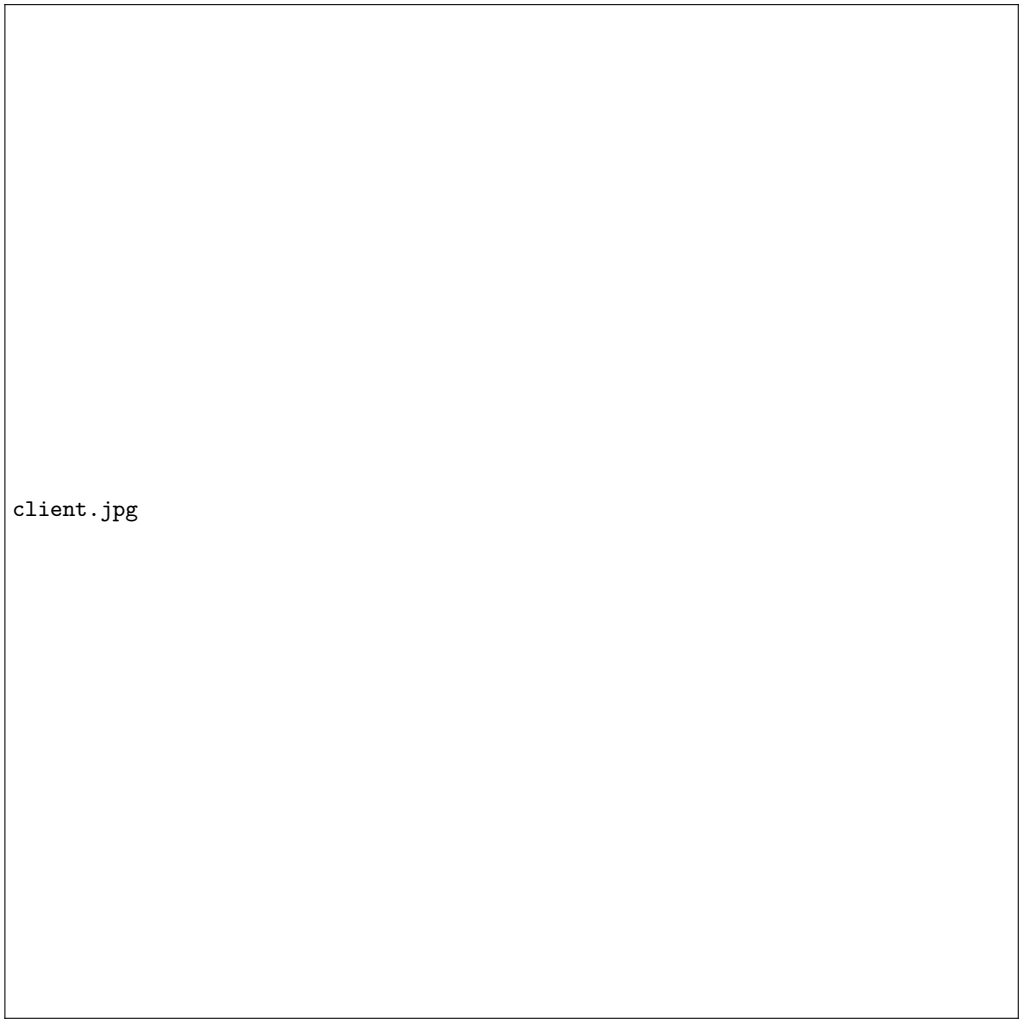

Fig. 2. OPC UA Client Structure[**?**]

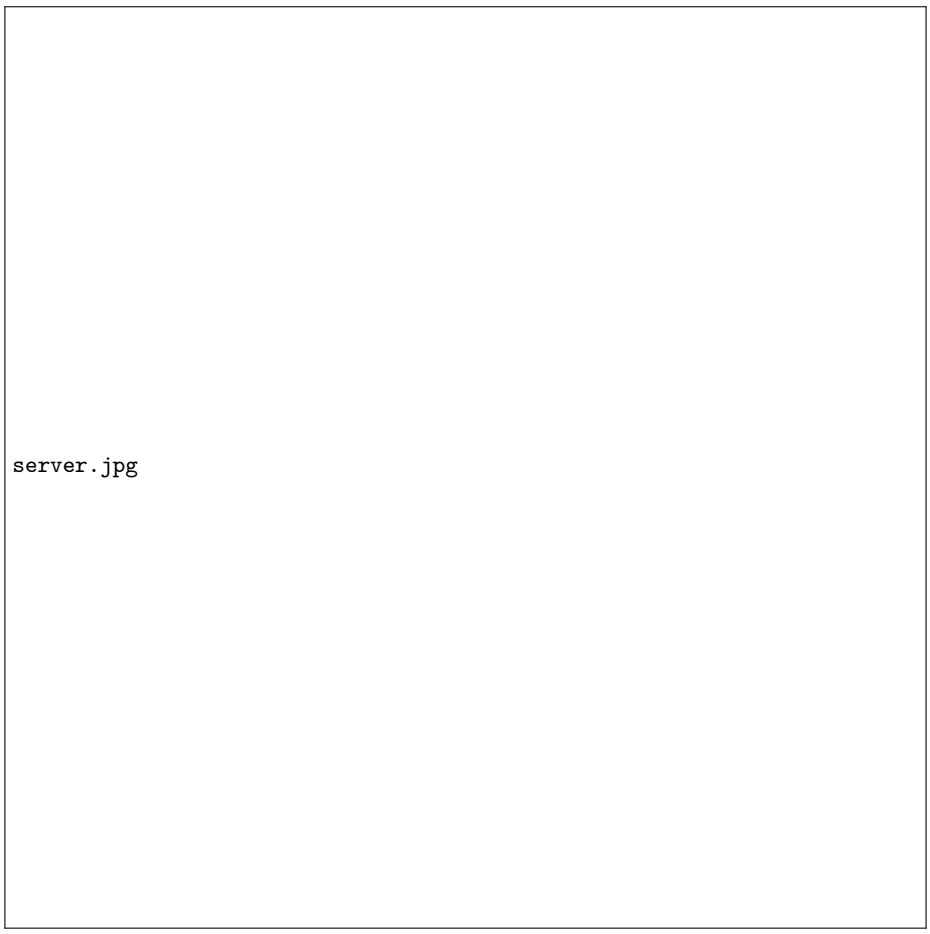

Fig. 3. OPC UA Server Structure[**?**]

level of the application to meet the need of reality. In the security model, authentication of client and server, authorization, integrity and confidentiality of client-server- communication, auditability and availability of services are guaranteed. Also OPC UA provides a set of countermeasures against message flooding, eavesdropping, message spoofing message alteration, message reply, server profiling, session hijacking and so on[**?**].

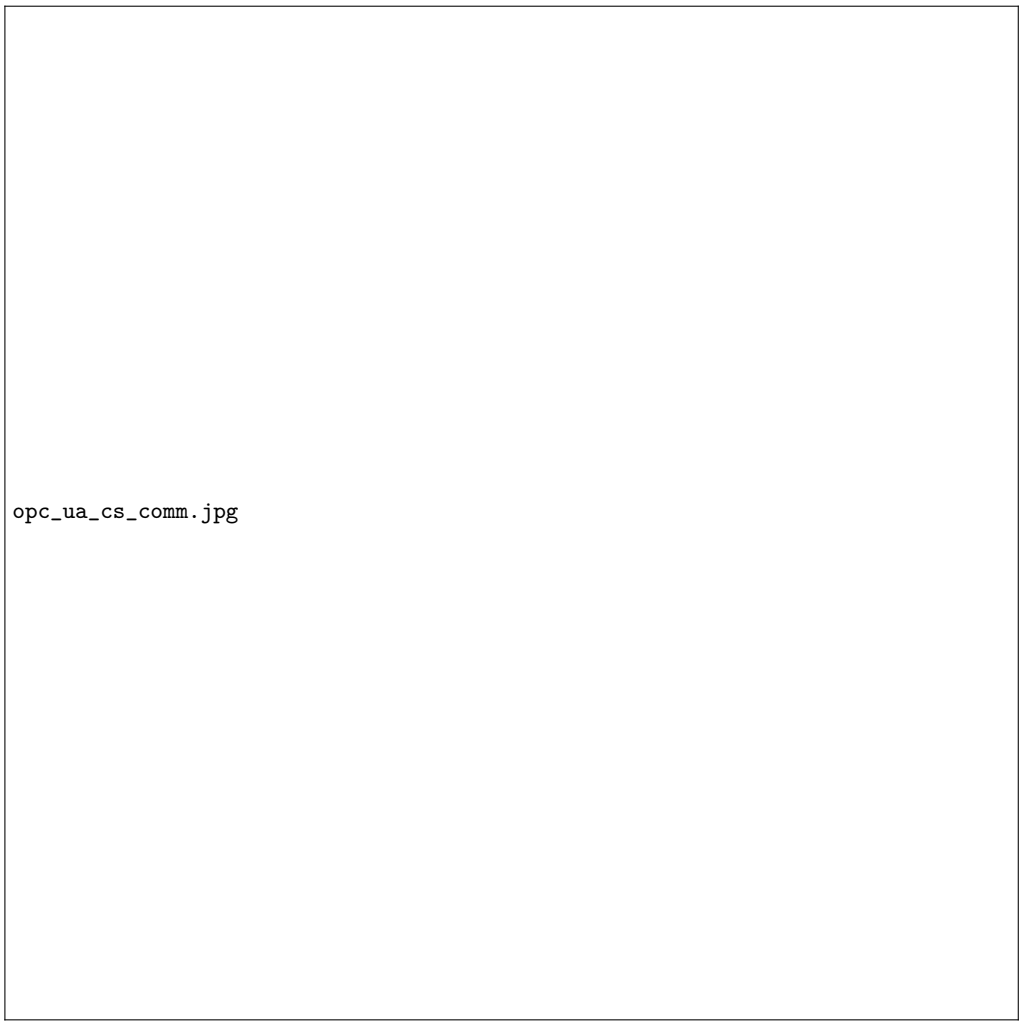

Fig. 4. OPC UA Client Server Communication[**?**]

Figure 4 pictures the typical security communication architecture of OPC UA. As shown in 4, the communication between OPC UA client and server is

established above a secure channel, which is active during the whole application session and in this session, the state information, such as subscriptions from client, user credentials, is maintained. The secure channel is established only after successful validation of both client and server certificates and it provides necessary mechanisms to support confidentiality, message integrity and application authentication. On top of secure channel, is an application level session between OPC UA client and server, whose responsibilities are to transmit data information and commands. This session is also in charge of managing security policies like user authorization and authentication. It should be pointed out that, even a secure channel is out of work for some reasons, the session is still valid and OPC UA client and server involved in aforementioned session can still re-establish the broken secure channel. A secure transport layer is guaranteed by encryption and signatures methods provided by platform that supports OPC UA structure.

Security Handshake Security handshake as below explains with some details about how secure channel and session are established.OPC UA client initiates the first *OpenSecureChannel* request and waits the response from server. Messages exchanged during the process of construction secure channel between client and server are encrypted using asymmetric encryption and signature algorithms. But some security protocols that could be applied according to OPC UA standard, are not using an asymmetric message encryption algorithm to encrypt to request/response messages. Instead, they apply AsymmetricKeyWrapAlgorithm to encrypt symmetric keys and use symmetric encryption algorithm with encrypted keys to encrypt messages. After a successful construction of secure channel, OPC UA client sends *CreateSession* request and waits for server response. Messages transported during this procedure are encrypted with symmetric encryption algorithms and signed with client/server signing key.

3.4 OPC UA Communication stack

As discussed in subsection 2.3, the OPC UA communication stack is a three-layer architecture: application layer, communication layer and transport layer. Even the terminologies of those layers are defined as the ones used in ISO model, but layers in OPC UA are not directly equal to layers in ISO model. Figure 6 from OPC UA 6th specification[**?**] gives a precise overview of each layer in OPC UA communication stack model, meanwhile it demonstrates functionalities performed by each layer.

UA Application layer realizes client and server code. Serialization layer together with secure channel layer build the communication layer and their job is dividing long message into pieces referred as message chunk, encrypting each individual message chunk, not entire whole message and forwarding encrypted message chunk to transport layer. When receiving message chunk from others, OPC UA message receiver firstly verifies whether this message piece meets the security standard negotiated between OPC UA client and server. If not, this

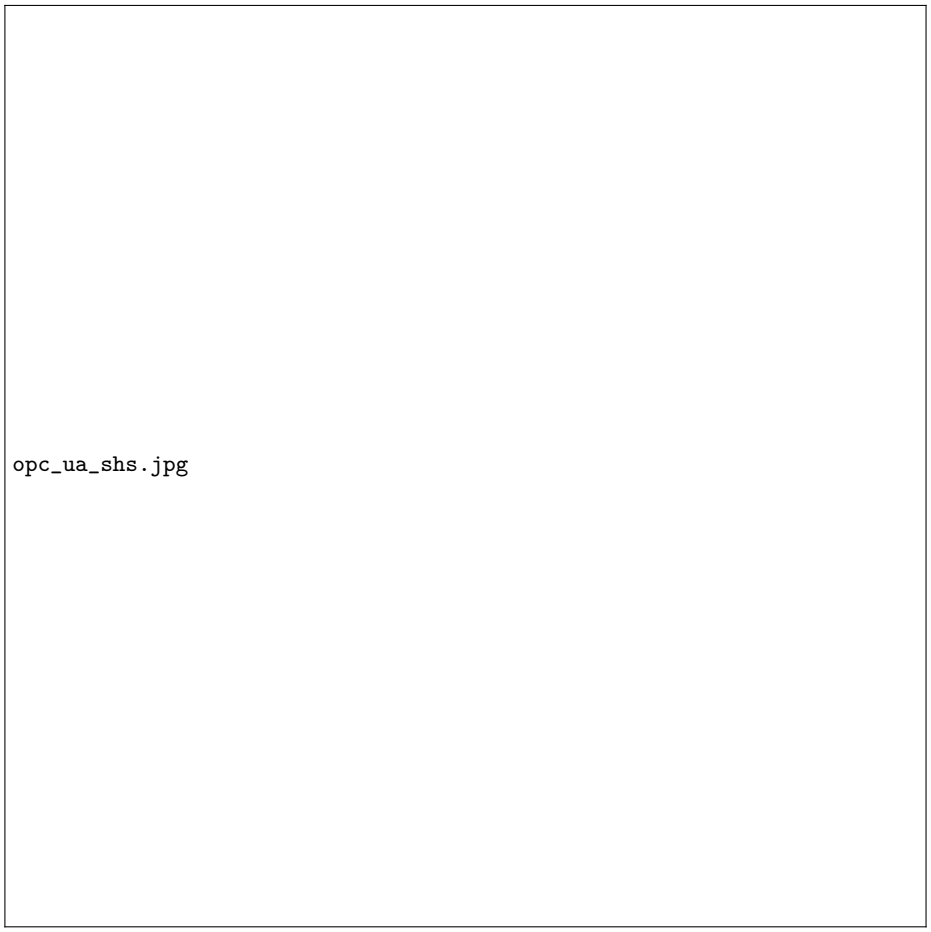

Fig. 5. OPC UA Client Server Security Handshake[**?**]

message receiver will close the secure channel. After a successful verification of all message chunks, the original OPC UA message will be reconstructed and sent to UA Application Code through API. Each secure message chunk applies the following structure described in figure 7.

Knowing the essential parts of OPC UA communication stack, in the following subsections the establishment, re-establishment and close of communication channel are explained.

Establishment of communication channel As the first step to create TCP/IP connection, this process is always initialized by OPC UA client. OPC UA client initiates his socket and sends *helloMessage*, that includes supported buffer size which specifies the message chunk size used for future communication, to the tar-

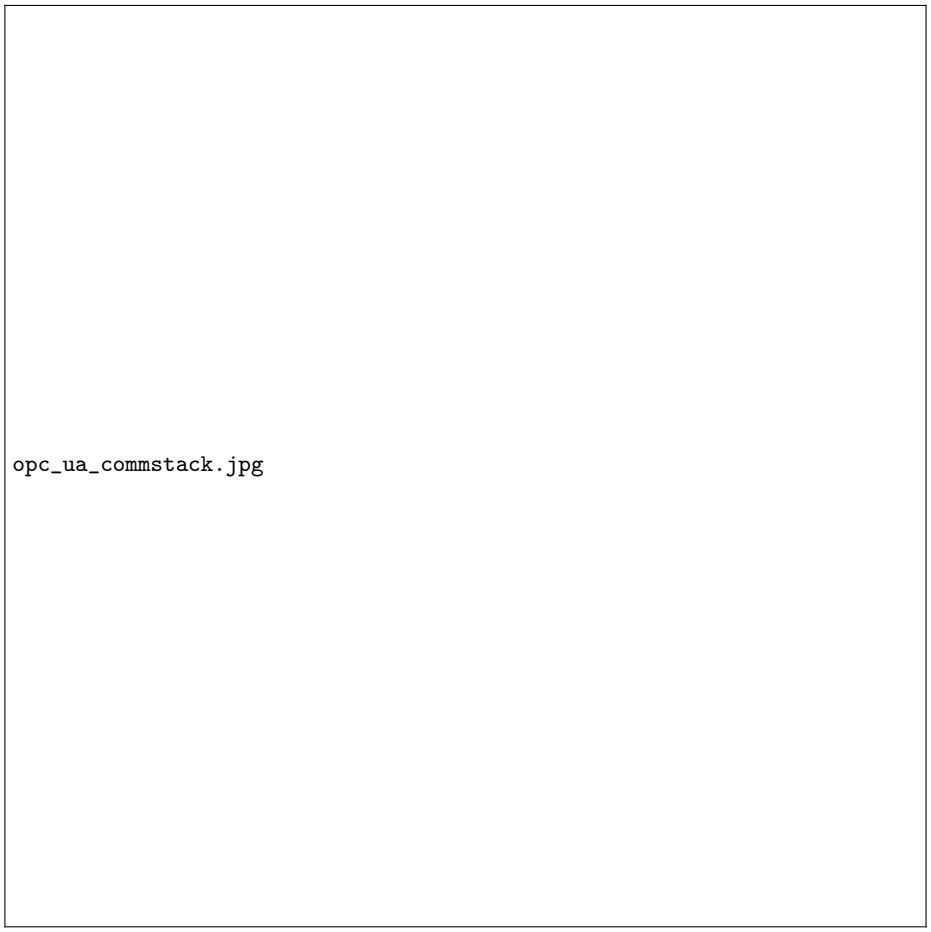

Fig. 6. OPC UA Client Server Communication Stack[**?**]

get OPC UA server. After receiving greeting message, OPC UA server answers the request for establishing TCP/IP connection with acknowledge message and reports negotiated buffer size to his own secure channel layer. Moreover during the creation of communication channel process, the greeting *hallo* and answering *acknowledgment* messages could only be sent once. If OPC UA client or server receives them more than one, error will be reported and corresponding communication socket will be closed. Even thought server application code does not have to work during the negotiation of secure channel process, it should provide the communication stack all his trusted certificates which help communication stack to verify the identity of the other communication partner.

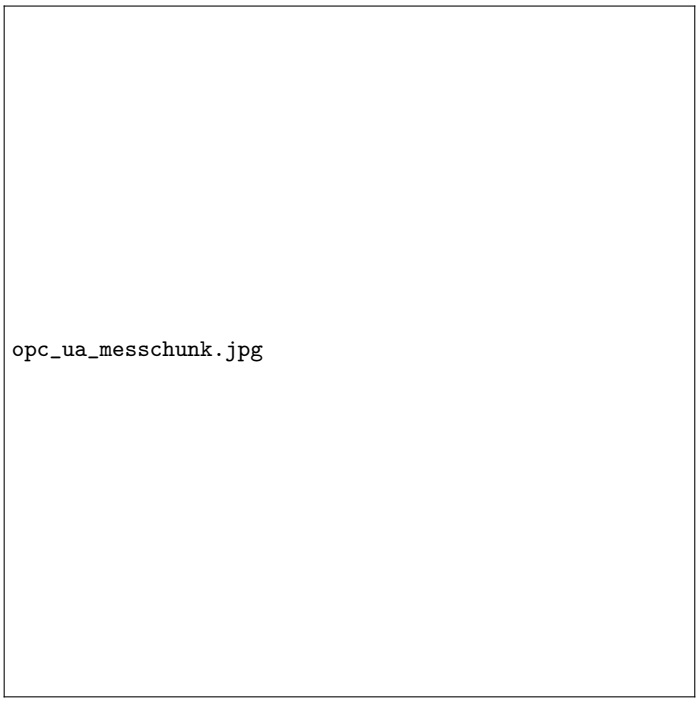

Fig. 7. Message Chunk Structure[**?**]

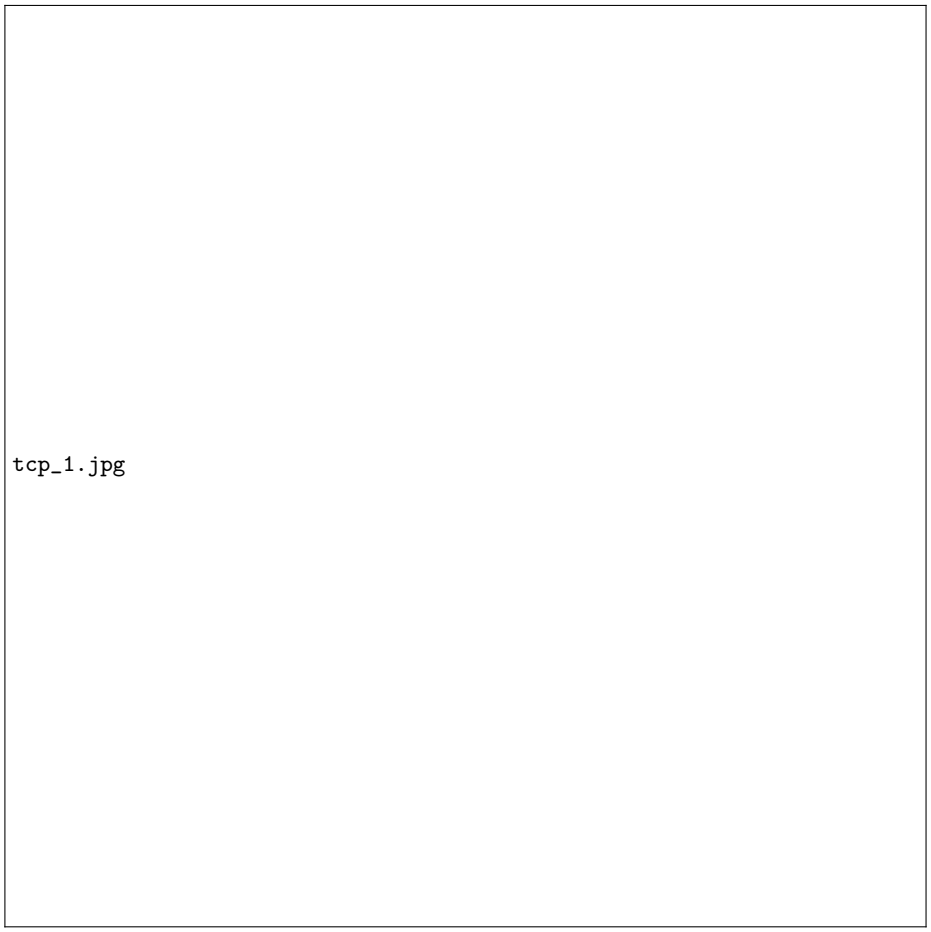

Fig. 8. Establish TCP/IP Connection[**?**]

Close TCP/IP Connection This process is done when OPC UA server receives *CloseSecureChannel* request from OPC UA client. During this process, server releases all the resource taken by corresponding secure channel and sends none response.

Recover Secure Channel Whenever error occurs during TCP/IP connection between OPC UA client and server, client will try to periodically re-establish it until the session is closed or the lifetime of security token goes to an end. Also it should be pointed out that the buffer size defined by corrupt secure channel should not be changed during this error recover process.

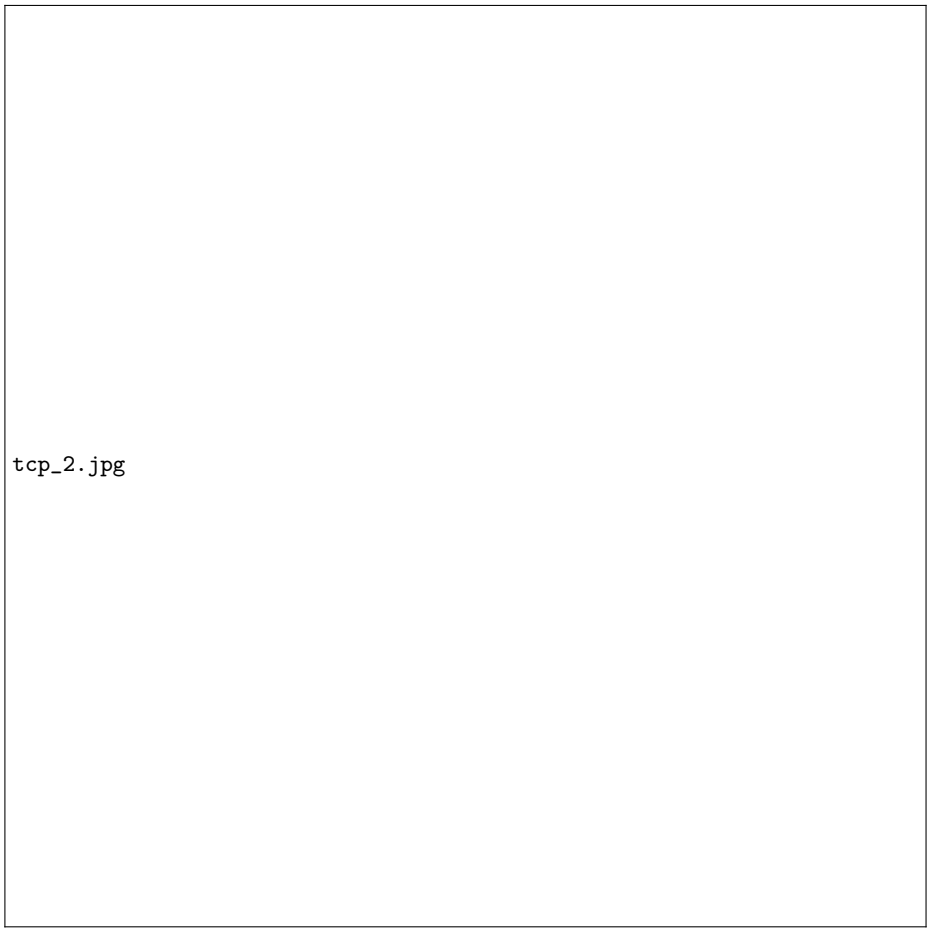

Fig. 9. Close TCP/IP Connection[**?**]

3.5 Historical Data

Last but not least security feature offered by OPC Unified Architecture is auditing, which supports traceability of any behaviours occur in OPC UA system. That means any security related problem can be recorded and for future use.

3.6 Other Competitor

WebSphere Message Broker Message Queuing Telemetry Transport (MQTT)[**?**] is another machine to machine (M2M) communication protocol. Compared with OPC UA standard, MQTT also supports UDP protocol in the transport layer. In OPC UA, only unidirectional, client to server, communication is provided, but in MQTT server to client communication is also possible without server

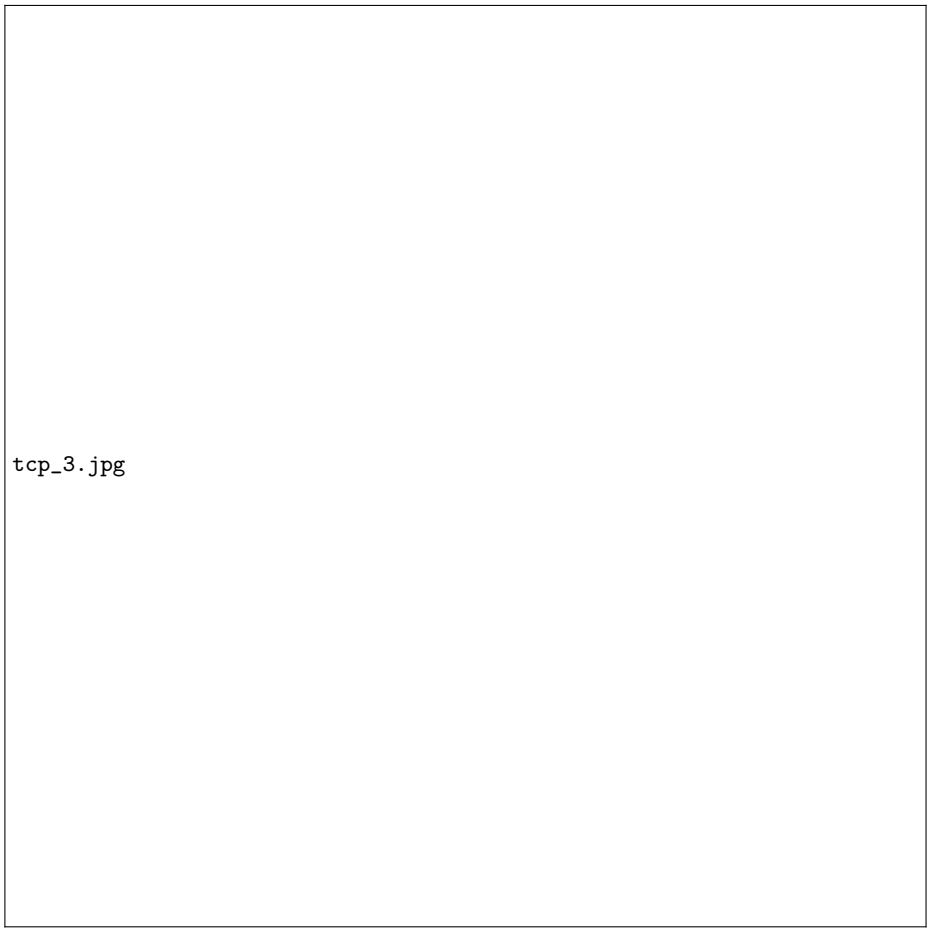

Fig. 10. Recover Secure Channel[**?**]

implements client code. Moreover the communication overhead of MQTT is in comparison with OPC UA is relative small.

Even thought MQTT protocol supports communication environment with low bandwidth and high latency, OPC UA provides complex object model and supports more features, including historical data record, alarm, notification, complete security policies and this is reason why OPC UA is more suitable for the application scenario that handles sensitive data with complex structure and needs immediate response.

Another member from Internet of Things is Constrained Application Protocol (CoAP)[**?**] which is designed for the extreme simple electronic devices with less memory and computing power and original CoAP only runs over UDP. Compared with OPC UA, simplicity from CoAP is the advantage, but apparently it should be considered that in the implementation scenario other transport proto-

col could be used, like TCP, more functions and services other than pure message exchange between client and server, are requested from users.

4 Implementation Scenario

Fig. 11. Smart Home

Figure 11 describes the basic structure and functionalities of Smart Home. Central Controller also named as OPC UA Server is in charge of monitoring predefined environment variables, for instance, home temperature, luminance and how much water the pet has, and taking corresponding behaviors, such as opening windows, turning off the heating or notifying pet owner that puppy needs water. With the help of such services a more comfortable living condition is created in an automated way. Also OPC UA Server controls access right of entering each room, which means only authenticated users with enough authority can open the door. Moreover the root user, namely the owner of this house, is capable of assigning the permission of accessing particular room to other guests. In case of when he/she is taking a vocation and pet cannot get necessary care while he/she is away. At the same time, root user won't worry about the person, who promises feeding pet, entries the forbidden room. At last, every single action taken by the OPC UA Server is recorded and is available for future use.

In the implementation scenario, OPC UA clients are Universal Integrated Circuit Card (UICC) based phone user and in home allocated senor. OPC UA server is a secure hard device, that is in charge of communication management with phone use and implementation of OPC UA server application code. Environment date is measured periodically by sensors and each room is locked, only the client, who is authenticated by server and holds enough authority can enter. It is assumed that handy users are in an open environment at mean while senors in home have a relative secure wifi connection with OPC UA server. Smart card that is applied in this scenario also acts as security token for OPC UA client, which contains credential information like encryption keys. certificates and digital signature. Moreover the communication stack is developed and integrated on smart card, which means without corresponding UICC card, OPC UA client and server are not able to appropriately finish their work.

Fig. 12. OPC UA Client Server Based On TCP/IP or APDU

Figure 12, describes possible software structure of aforementioned OPC UA client server structure. With different chip card, OPC UA client application is able to communicate with server using Application Protocol Data Unit(APDU)

and Short Message Service(SMS), which is a more nature and traditional way to exchange date with chip card, or TCP/IP based web service when components from Global Platform[1] is applied or newly released Javacard 3 is used.

4.1 Software Structure

According to the above-mentioned application scenario, UA application client and server code will be written using Java, deployed as Handy application, OTA server respectively. Basic functions as following are provided:

- subscription/publishing environment data
- secure message exchange
- authority management
- historical data management
- running client's command

Communication stack is develop on UICC smart card realizing secure channel and session management, transporting data to message chunk receiver using Tcp/IP connections. An internal API translates OPC UA application instructions in to Application Protocol Data Unity (APDU) messagse and forwards them to smart card, which is in charge of user authentication and processing secure messaging between card application and chip card pair.

Smart Card Security As explained, smart cards are widely used in applications that require strong protection. With sophisticated communication protocol using Application Protocol Data Units, smart card and Card Accepting Device(CAD) are able to process secure message exchange and bidirectional authentication. Moreover sensitive data like certificates, encryption keys are stored on card along with other precious user information and this data is extreme difficult to be altered by third party .Most of time smart card also acts as secure token and can process cryptographic algorithms on hardware. Nowadays, smart card supports symmetric key algorithms like DES, triple DES; standard public key cryptography for instance RSA, hash functions such as commonly SHA-1[**?**]. More powerful microprocessor on chip card is, the speed performance is better.

Moreover thanks to self-containment structure, smart card itself does not dependent on other external resources, which could be extreme vulnerable to potential secure attack, and therefore provides a better hardware security and OS security.

Fig. 13. Client Structure

[1] Global Platform is a cross industry, non-profit organization that develops and publishes standard in secure chip technology.

Client Structure As described in figure 13, the OPC UA client consists of client application code that realizes client application level functions, OPC UA client API that translates client application instructions into APDU and forwards APDU to UICC smart card. The Communication stack is developed and integrated with UICC card, which is in charge of creation and management TCP/IP connection, secure channel between client and server. This communication stack is based on card OS components provided by Global Platform. Global Platform provides and defines communication flow between an application provider and smart card, allows information exchanged between a remote entity and a card. The on card component, which is responsible for connection creation with the remote entity, is called Security Domain. And the remote entity also is referred as Remote Administration Server. With those concepts, smart card with Security Domain can act as HTTP client and is capable of packing APDU formate information into HTTP POST message. At the same time, the Remote Administration Server also is a HTTP server, which can send HTTP message including APDU format information to its client.[**?**]

Figure 14 illustrates a typical communication flow between administration server and corresponding security domain on smart card. As can be seen, the request for open communication is alway initialized by security domain, which is also the phone user. After a successful creation of secure handshake, the remote administration server and security domain is able to using HTTP message to exchange command and response strings, which include APDU instructions from OPC UA client and server.

Server Structure The server structure for Smart Home is pictured as figure 15 and it consists of an OPC UA server application that implements smart home server services, a remote application server, which realizes server side communication stack, together with on card embedded security domain manages HTTP connections, secure channel construction as well as user authentication.

5 Time Lines

- read paper, documentation, reference
- design dummy client/server application code
- analyze and design communication stack that fits UICC card and meets OPC UA standard
- combine application code and communication stack
- analyze design and apply secure protocols.
- debugging/performance analysis

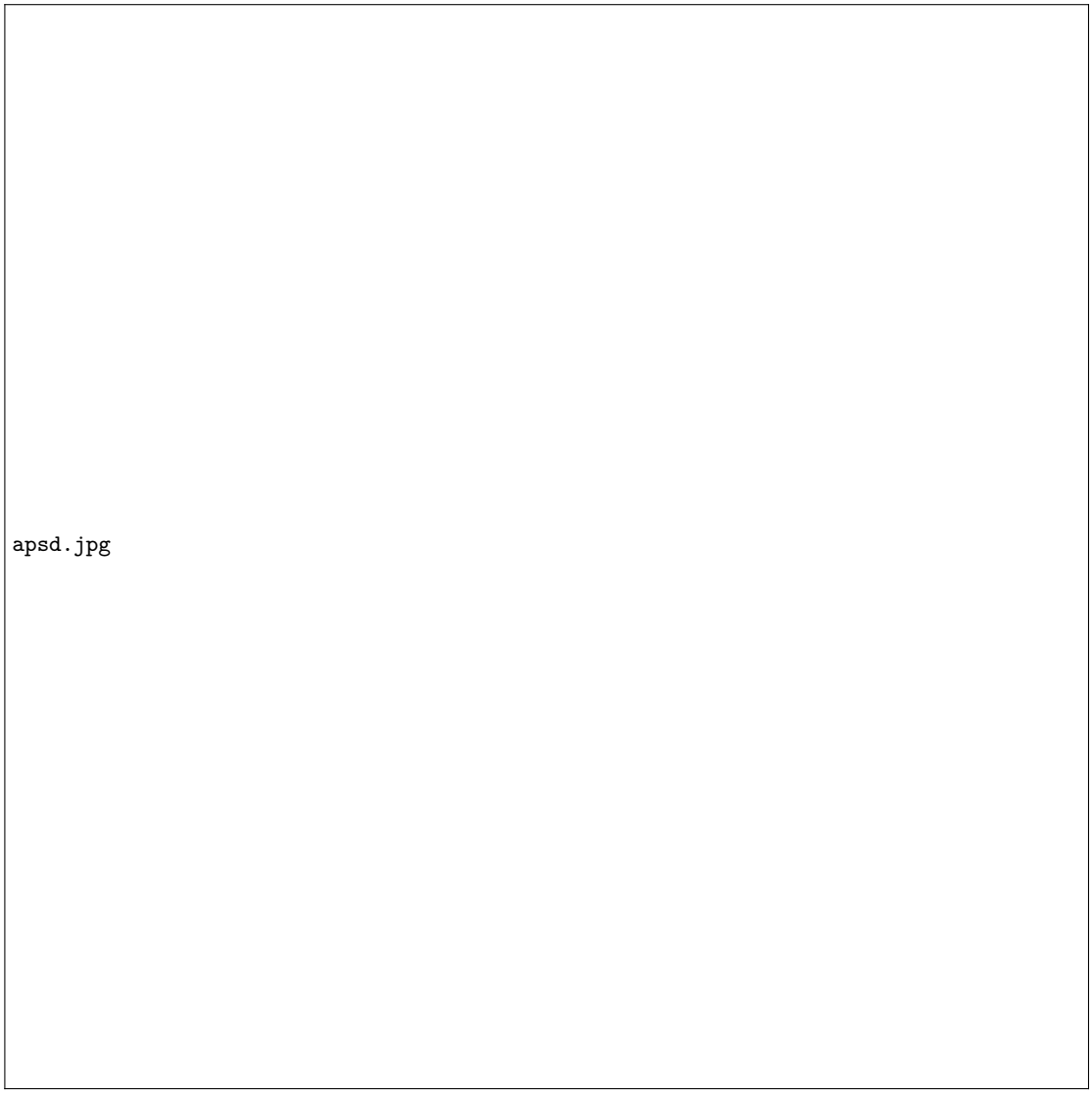

Fig. 14. Communication Flow between an AP and corresponding APSD[**?**]

Fig. 15. Server Structure