Abstract API Parlay-X Web Services

Technology Reports

# SCN: Providing Service Scenario Execution Infrastructure Using Abstract APIs —Koe-no-Takuhaibin—

*Conventional telecom network services have been implemented with detailed control over complex call state transitions. As a result, it has been difficult to reuse processing or functionality, even when it could be shared among multiple services, and this difficulty of reuse has resulted in higher development costs. To resolve this issue, reusable functions are isolated in the form of abstract APIs and SCN was developed as a service scenario execution infrastructure in order to control them. We also developed the Koe-no-Takuhaibin (Voice Delivery) Service, the first service built on SCN, contributing to the diversification of voice-communication services.*

Core Network Development Department

***Kiyotaka Inoue***
***Masaru Takahashi***†
***Yu Kojou***
***Aya Tsurusaki***

## 1. Introduction

NTT DOCOMO has always strived to find ways that it, as an operator, can provide added value to communication between its users. With the recent spread of smartphones and the increasing number of client server applications that use an internet player (networks increasingly becoming "dumb pipes"), it is becoming more and more important for operators to find added value that only they can provide. Now, NTT DOCOMO has introduced the new Service Composition Node (SCN), with the goal of implementing such services quickly and cost-effectively on the core network [*1], which has been converted to the IP-Multimedia Subsystem (IMS) [*2]. With the development of SCN, we are introducing technology with the latest architectural developments, including Service Oriented Architecture (SOA) [*3] and the Java® platform Enterprise Edition (JavaEE) [*4]. These technologies enable development of services with abstract Application Programming Interfaces (APIs) [*5], providing cost benefits by making it easy to reuse functionality.

NTT DOCOMO has also developed the first service using SCN, the Koe-no-Takuhaibin (Voice Delivery) Service. Koe-no-Takuhaibin is a consumer service aimed at the approximately 50 million FOMA users, and is advanced in providing a large-scale commercial telecom service using abstract APIs, even from a global perspective.

In this article, we give an overview of the technology and equipment introduced with SCN, and describe the call-control methods used for the Koe-no-Takuhaibin service.

## 2. Issues with Conventional Development and Solutions through External Technology Introduced with SCN

### 2.1 Issues Conventional Development Methods

Two issues that have been raised



† Currently Planning & Coordination Office

*1 **Core network**: A network consisting of switches and subscriber-information management equipment. Mobile terminals communicated with the core network via the radio access network.

*2 **IMS**: IP-based core network specification standardized by the 3GPP. Communications are implemented using protocols including SIP (see *14) and Diameter (see *23).

*3 **SOA**: One possible architecture for IT systems. Functionality is allocated with a focus on services (business-level transactions), allowing reusability of completed systems to be improved.

with conventional development methods for core network services are that programs have not been very reusable, and that functionality not directly related to the service requirements, such as maintenance and infrastructure functions, had to be developed separately for each service.

1) Development of Programs with Low Reusability.

Programs tended not to be reusable because processing was specialized in detail to individual services, and Operators (as enterprises) continuously pursued low cost. As a result, implementations tended to be over-optimized. Further, node functions in conventional switch-based programs often could not be designed for use by other services or external nodes, because they were inherited from earlier versions, among other reasons. Thus the internal interfaces were often device-specific or bound tightly to the service, and functionality was not developed for more general use.

2) Development of Functionality not Directly Related to the Service Specifications (Maintenance, Infrastructure Functionality)

One of the primary factors increasing cost with conventional development methods was development of functions not related to the service specifications, together with the presumed system architecture. Examples of this include control transfer functions. Control transfer functions are used when the resources become insufficient in equipment handling a range of subscriber numbers, to expand resources by adding equipment and redistributing load over the new equipment without stopping services. Complex processing is required when transferring capacity from older equipment to new equipment, including mechanisms to synchronize data, and reroute signals from the older node. Costs increase dramatically due to these complexities, so there is need for improvement.

### 2.2 Resolving Issues Using External Technology

To resolve these issues, we introduced external technology for development of SCN. Conversion of the NTT DOCOMO core network to IMS has also increased compatibility for introduction of these technologies.

1) Developing Highly Reusable Programs with SOA and SDP[*6]

SOA is an architecture that has been a focus of development for highly reusable programs in the enterprise field. As our reference model for development of SCN, we used an architecture called Service Delivery Platform (SDP), which is an application of the SOA concept to the telecom field. SDP has also been called an added-value service-creation platform, and was created to build Web services from existing call services and services maintained by other telecom vendors, to combine them with other internal and external Web services, and to develop and operate new communications services.

2) Minimizing the Scope of Program Development with JavaEE

We adopted JavaEE 5 for SCN in order to minimize program development unrelated to the service specifications.

This allowed us to remove complexity and increase application reusability through pure object-oriented development. We were also able to minimize the scope of application development by using transaction-management functions maintained by the JavaEE platform, such as Enterprise Java Beans (EJB)[*7] and the Java Persistence API (JPA)[*8], service profile management functions, and message management functions such as Message Driven Beans (MDB)[*9], and the Java Message Service (JMS)[*10].

JavaEE also provides APIs that assume integration with Web services, which provides good compatibility with SDP, as described in 1).

Future telecom services will also need user configuration of service preferences (customization), and Web access as a means to check status. With adoption of JavaEE 5, Web applications can be developed using Java Servlets[*11], allowing provision of services that are more friendly than ever before.

---

*4 **Java®EE**: The name and specification for a server-side Java development and run-time environment. In earlier versions it was called J2EE 1.2, but since version 1.5, the name has been changed to JavaEE 5.
Oracle and Java are registered trademarks of Oracle Corporation, its subsidiaries, and affiliates in the United States and other countries. Company and product names appearing in the text are trademarks or registered trademarks of each company.

*5 **API**: An interface that provides functionality of an application program. Allows developers to develop programs by combining control logic with API calls.

*6 **SDP**: An architecture that applies the SOA concept to the telecom field, increasing the reusability of systems that are already built and allowing diverse telecom services to be offered by combining the systems.

*7 **EJB**: An architecture for program components defined in J2EE. Includes Session Bean, Entity Bean and MDB (see *9) components.

*8 **JPA**: An API specification for handling relational databases with Java.

### 2.3 Abstract APIs and Enablers

In this section, we describe abstract APIs and enablers, which are functional elements of SDP.

Abstract APIs conceal complexities such as protocol signal processing and processing of special functions, and can be controlled through simple processing of requests and responses. With SOA, entities[*12] that provide abstract APIs are called Web services, but in the remainder of this article we will call such entities enablers.

An overview of abstract APIs is shown in **Figure 1**. The originating User Agent (UA)[*13] call-processing enabler hides call processing for the Session Initiation Protocol (SIP)[*14], and provides an abstract API as specified in Part 2 of Parlay-X 3.0[*15] [1]. SIP call processing requires several signals to be processed, but in the service scenario in Fig. 1, call initiation can be controlled with a single call to the MakeCallSession API. The enabler also hides processing of various semi-normal and error cases during SIP call processing such as missed signals, timeouts, and parameter errors. These parts of SIP call processing are the same, even if the service is different. In this way, processing for such semi-normal and error cases no longer needs to be coded for each service scenario, even for new services, reducing the scope of development required.

Note that to implement service specifications requiring complex call control, cases requiring service-specific sequences may not be suitable for development using SCN, and conventional development methods using nodes must be used.

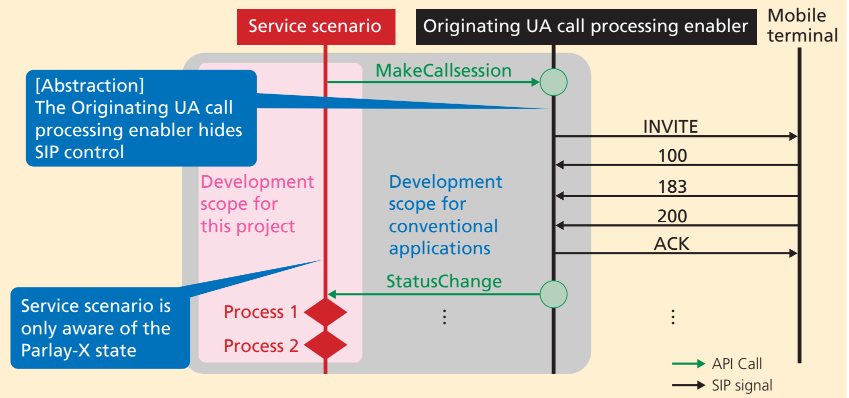


**Figure 1 Abstract API image**

### 2.4 Objectives in Introducing SCN

We introduced SCN with the objective of using external technology to resolve issues with conventional development methods. This allowed us to minimize development scope for each telecom service by converting telecom functions to enablers, to increase reusability by hiding complex processing with abstract APIs, and to utilize the increased development productivity provided by JavaEE.

## 3. SCN System Overview

The logical structure of SCN is shown in **Figure 2**. SCN is composed of the three logical functions: a service scenario, which controls the service by making API calls; enablers, which provide abstract APIs; and an SCN-DB[*16], which maintains service-specific user data.

The service scenario is an application program consisting of conditional branching logic and API calls, and is dependent on the individual service. For example, three different service scenarios would be required to provide three different services. On the other hand, enablers are independent of services. APIs are called from multiple different service scenarios and are shared by all of them.

This minimizes processing that is different and specialized to an individual service, isolates it and gathers it in the service scenario.

### 3.1 Overview of Logical Functions

1) Service Scenario

With SCN, the execution environment for the service scenario is called the SCN Application Server (SCN-AS), and is composed of the platform functionality provided by JavaEE, and the

*9 **MDB**: An EJB that enables asynchronous communication to be implemented easily, and enables asynchronous method calling when combined with JMS (see *10).

*10 **JMS**: An API specification defined in J2EE for sending and receiving data. Used for calling asynchronous methods.

*11 **Java Servlet**: A function of JavaEE. Provided to allow control of Java applications using HTTP and SIP (see *14) protocols.

*12 **Entity**: A structural element that provides functionality within a logical architecture.

*13 **UA**: An entity that sends and receives SIP (see *14) signals.

*14 **SIP**: A call control protocol defined by the Internet Engineering Task Force (IETF) and used for IP telephony with VoIP, etc.

*15 **Parlay-X 3.0**: Defined by the Parlay Group, an organization for standardizing telecom APIs. An API specification for developing applications in a Web services environment using telecom services.

*16 **SCN-DB**: The database within the SCN system.

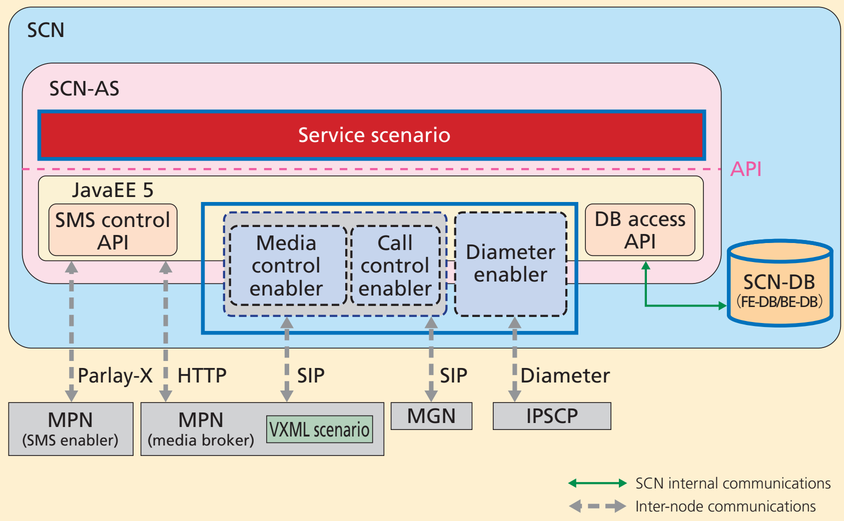


**Figure 2 SCN logical organization**

Service Scenario, which performs service-specific processing.

The service scenario is organized into three layers: a presentation layer, which has interfaces with enablers and the client, a business logic layer, which implements the service specification, and an integration layer, which interfaces components like the SCN-DB and the SMS enabler[*17].

Implementations in the presentation layer could include generating requests using Java Servlets, or Simple Object Access Protocol (SOAP)[*18] messages using the Java API for XML-Based Web Services (JAX-WS)[*19]. The integration layer contains implementations such as Object-Relational (OR) mapping[*20] with a Relational Database Management System (RDBMS)[*21] using JPA, or SMS asynchronous transmission processing using MDB/JMS. Then, the business logic layer does not depend on the presentation layer and can be reused. For example, logic created for voice-call processing can be reused as-is for a Web application.

In this way, the APIs provided by JavaEE are used for development of each of the layers, and JavaEE plays a major role in increasing development productivity.

2) Enablers

Enablers, providing abstract APIs, are also implemented in SCN, in addition to service scenarios. Enablers implemented in SCN include a call control enabler, which abstracts call processing and conducts SIP interworking[*22] with the IMNS network based on 3GPP standards (iSC-interface); a media control enabler, which abstracts interworking with media processing equipment; and a Diameter enabler, which abstracts Diameter[*23] signal processing with the Home Subscriber Server (HSS)[*24] based on 3GPP (Sh-interface) specifications.

The call-control and media-control enablers in SCN have a tightly linked relationship, and they provide an API that is a mashup[*25] of call control and media control.

3) SCN-DB

The database in the SCN system is called SCN-DB. SCN-DB consists of the Back End DB (BE-DB), which maintains the service data persistently, and the Front End DB (FE-DB), which maintains the session data for individual calls.

A disk-based DB is used for BE-DB because it maintains a large volume of data, requires very strong fault tolerance, and is accessed relatively infrequently.

Conversely, an in-memory DB[*26] is used for the FE-DB, because data volume is relatively small and data is accessed frequently, requiring millisecond response times. Also, by storing session data on external equipment, SCN-ASs implementations can be stateless[*27] and N-Act[*28].

### 3.2 Objectives in Allocating Functionality to Logical Functions

As described above, no data is maintained in the SCN-AS, which performs service control processing, in SCN. Data is stored on two physically-

*17 **SMS enabler**: An enabler that provides the abstract APIs for SMS control, as specified in Parlay-X 3.0 Part 4.

*18 **SOAP**: A protocol for remote program calls. Generally uses HTTP as the lower-level protocol and exchanges XML documents.

*19 **JAX-WS**: A Java API specification for remote program calls.

*20 **OR mapping**: A mapping between objects and a relational database. Simplifies DB access in Java programming.

*21 **RDBMS**: A database management system that manages data based on tables with multiple rows and columns. SQL is used to access the database.

*22 **Interwork**: Interaction between communications systems.

*23 **Diameter**: An extended protocol based on the Remote Authentication Dial In User Service (RADIUS), and used for authentication, authorization, and accounting in IMS.

*24 **HSS**: A subscriber information database in a 3GPP mobile communication network; it manages authentication information and network visiting information.

separate devices: the BE-DB, which stores permanent data, and the FE-DB, which stores session data. Therefore, the BE-DB and the FE-DB can be expanded independently, increasing the amount of session data if the load increases due to increased traffic, or increasing the amount of permanent data if the number of users increases. The BE-DB, which stores the permanent data, also performs data redistribution when storage is added, but execution of this operation is hidden from the service scenario by the DB functions, so data migration programs do not need to be developed, and no migration work is required in operation.

### 3.3 Use of APIs External to SCN

The Koe-no-Takuhaibin Service, which was the first service built using SCN, required functionality to send SMS messages, so we developed an SMS enabler external to SCN. This involved making an enabler for an existing node, so we added functionality to the Media Processing Node (MPN)*29, which provides SMS Center (SMSC)*30 functionality, with the "Short Messaging" abstract API[4] as specified in Parlay-X 3.0 Part 4.

As a result, how SCN service scenarios can provide services by combining abstract APIs, regardless of whether they are internal or external to SCN.

## 4. Koe-no-Takuhaibin Service

The Koe-no-Takuhaibin Service is a service that provides non-real-time voice communication through a Koe-no-Takuhaibin Center within the NTT DOCOMO network.

### 4.1 Service Features

This service has the following features, in comparison with conventional e-mail and telephone.

- The other party can be contacted by voice, regardless of their circumstances
- No text input is needed, so it can appeal to a relatively older range of users who are not accustomed to operating devices
- A playback notification function can inform the user when the message has been played back
- More personable messages can be sent than using e-mail, because they are transmitted by voice

This service is available to all FOMA voice subscribers, and can be used without requiring any application procedures.

### 4.2 Functions Implemented by the Service

1) Message Recording

The message sender connects to the Koe-no-Takuhaibin Center by dialing the special number, " (*2020) + receiver's phone number" (**Figure 3** (1)). After connecting, the sender records a message according to voice guidance. After the message is recorded, a recording-notification SMS message is sent to the receiver, including the special number,

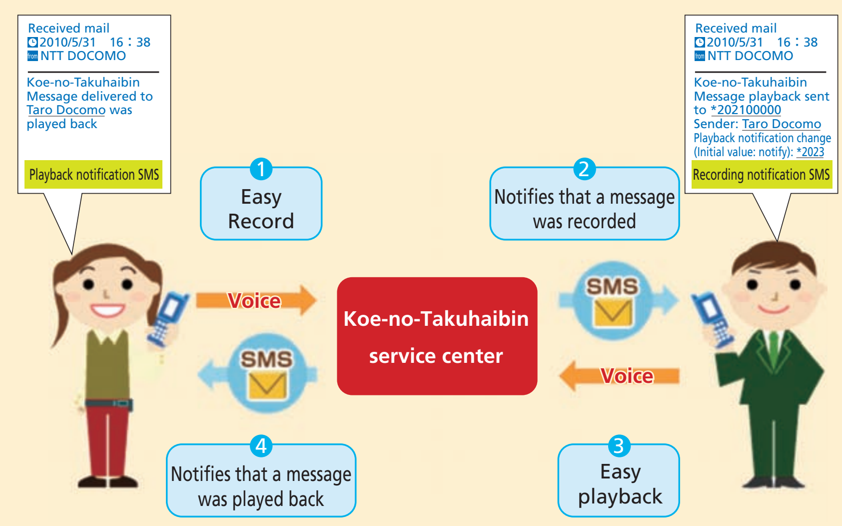


**Figure 3 Koe-no-Takuhaibin Service overview**

*25 **Mashup**: To create and provide a service by combining the content and services from several other, different services.

*26 **In-memory DB**: A database with accelerated access response, accomplished by expanding and maintaining the data in memory.

*27 **Stateless**: Indicates that data and state are not maintained in the system. Processing is done based on the input signal and access to a physically separate database, so that there is no need for synchronizing data among servers when redundant implementations are used.

*28 **N-Act**: Distribution of processing load over N servers running in parallel. If one of the servers fails, the other servers can continue processing.

*29 **MPN**: A node of the NTT DOCOMO core network. It provides state for various media services including voice answering, melody call, and other voice media services; and video media services such as videophone answering.

*30 **SMSC**: the SMS Center server, as standardized by the 3GPP. Stores and retransmits SMS messages.

"(*2021) + message ID" (Fig. 3 (2)). Note that feature phones[*31] sold after April 2011 are equipped with an easy calling method activated from the address book screen, so use of the special number is not required.

2) Message Playback

The receiver is able to play back the recorded message (Fig. 3 (3)) by calling the number given in the recording notification SMS message. After the message has played back, the receiver is able to perform various operations by following voice guidance, such as deleting the message or replying to the sender (recording a Koe-no-Takuhaibin message to the sender). Calling the number "*2021 + message ID" plays back a single message, but there is also a batch playback number (*2022), which can be used to listen to all recorded messages at once.

When a message is played back for the first time, a playback notification SMS message is sent to the sender, notifying that the sender's message has been played back (Fig. 3 (4)).

Note that in principle, messages are automatically deleted 720 hours after they are recorded, but up to five messages can be kept indefinitely using a save operation.

3) Stop/Start Operation

This service is provided to FOMA voice subscribers without any contract information, so it is initially configured to enable use of the service. The service can be started or stopped by calling the configuration special number (*2023). The sending of playback notification SMS messages to message senders can also be enabled or disabled. The various SMS messages sent by the Koe-no-Takuhaibin service are summarized in **Table 1**.

## 5. Koe-no-Takuhaibin Implementation

### 5.1 Function Allocation

The architecture and distribution of functionality of the NTT DOCOMO network are shown in **Figure 4**. On the International Mobile Telecommunication (IMT) networks[*32] and Circuit Switched IP (CSIP) networks[*33], the decision to connect to SCN is made based on special numbers (*2020 to *2023) entered by the user (Fig. 4 (1)). Information indicating that it is a connection to SCN is attached, and the connection is made accordingly. SCN only supports SIP, so the signal (ISDN User Part (ISUP)[*34]) from the IMT network is converted to SIP by the Media Gateway Node (MGN)[*35] (Fig. 4 (2)). Multiple SCN addresses are registered in the MGN, and the MGN determines which SCN to connect to in round-robin[*36] fashion.

In the SCN, the connect request from the MGN is first received by the call control enabler (Fig. 4 (3)). The call control enabler terminates the SIP, and after determining that it is a recording call, notifies the SCN-AS to start a session using the API (Fig. 4 (4)).

The SCN-AS receives the notification from the call control enabler, and if the special number entered was *2020, for example, it starts a recording scenario. The recording scenario first accesses the SCN-DB to determine the service usage state for the user, for example, checking the number of recorded messages (Fig. 4 (5)). Then, it invokes a Diameter enabler API to retrieve contract information, to determine factors such as family discounts and call blocking (Fig. 4 (6)), obtaining the information from IPSCP[*37] (Fig. 4 (7)). The recording scenario determines conditions such as conflicts and call

**Table 1 Koe-no-Takuhaibin SMS list**

| | |
|---|---|
| Recording notification SMS | SMS message notifying the receiver that a message from the sender has arrived |
| Playback notification SMS | SMS message notifying the sender that the receiver has played back the message |
| Protection notification SMS | SMS message notifying that when protecting a received message, protection has completed |
| Welcome notice SMS | SMS message giving an overview of Koe-no-Takuhaibin to the receiver when the receiver receives the first message |
| Inbox full notification SMS | SMS message informing the receiver that they are over capacity when the number of received message exceeds the maximum (500 messages) |

*31 **Feature phone**: Refers to conventional i-mode standard functionality, in contrast to smart phones.

*32 **IMT network**: Refers to NTT DOCOMO's 3G core network, built with common channel signaling No.7 and ATM (see *41) lines.

*33 **CSIP network**: Refers to NTT DOCOMO's 3G core network, converted to IMS. Used in contrast to IMT network.

*34 **ISUP**: A part of the Signaling System No.7 (SS7). A protocol used for control of public switched telephone networks, and handles connection processing in ISDN.

*35 **MGN**: A node in the NTT DOCOMO core network which performs conversion between ISUP and SIP, as well as between IP and Synchronous Transfer Mode (STM).

*36 **Round-robin**: One of the techniques of load distribution in networks. A number of devices capable of performing the same function are prepared and the requested process is allocated to them in turn.

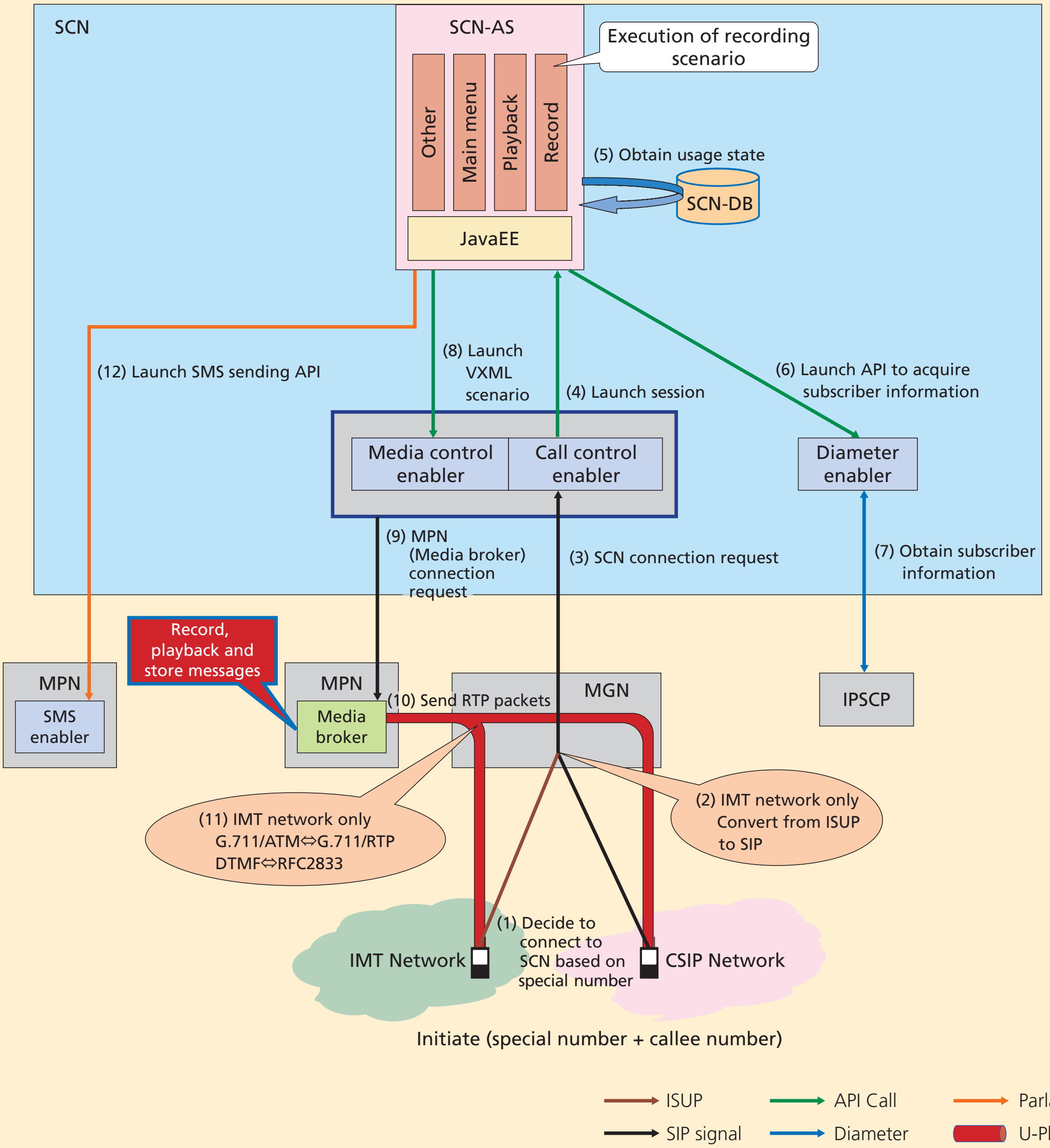


**Figure 4 Koe-no-Takuhaibin implementation**

charges from the contract information, and then launches a VXML[*38] scenario with the call control enabler and media control enabler (Fig. 4 (8)). The mashup API for call control and media control is launched, a connection to the media broker[*39] (MPN) is requested in order to connect in the U-plane and send guidance for recording (Fig. 4 (9)), and sessions are established between SCN and MGN, and between SCN and the media broker.

The MPN controls the appropriate VXML scenario, sending Real-time Transport Protocol (RTP)[*40] packets to

*37 **IPSCP**: A device in the FOMA network/PDC network that integrates the functions of both NMSCP that accumulates subscriber information and Mobile MultiMedia service Infrastructure (M3In) that provides mobile multimedia services, and manages advanced services within future core networks.

*38 **VXML**: An XML-based language used for automatic answering and other systems. Allows interactive application structures to be described in XML. Defined by the World Wide Web Consortium (W3C).

*39 **Media broker**: A logical node on the NTT DOCOMO core network that provides voice and video multimedia services in coordination with SCN. Terminates RTP (see *40).

*40 **RTP**: A real-time multimedia transport protocol via IP networks defined by the Internet Engineering Task Force (IETF).

the MGN (Fig. 4 (10)). If the sender is in the IMT network region, the MGN converts to ATM[*41]. For the Dual-Tone Multi-Frequency (DTMF)[*42] signals sent when the user performs push-button operations, IMT and CSIP network formats are different, so the MGN converts to a format supported by SCN (RFC2833[*43])(Fig. 4 (11)).

When the user has finished talking, the scenario receives notification through an API from the call control and media control enablers, and invokes the SMS sending API of the MPN (SMS enabler) to notify the receiving user that there is a recording (Fig. 4 (12)). Parameters such as the destination phone number and SMS text can be specified through the API, and the MPN (SMS enabler) sends an SMS message with the desired text to the appropriate user, as instructed.

## 5.2 Processing Overview

The sequence for an example of recording with the Koe-no-Takuhaibin Service is shown in **Figure 5**.

Heartbeat monitoring is done in the User Data Plane (U-Plane)[*44] RTP Control Protocol (RTCP)[*45] between the MGN and media broker (MPN), so the call state in both C-Plane and U-Plane can be detected.

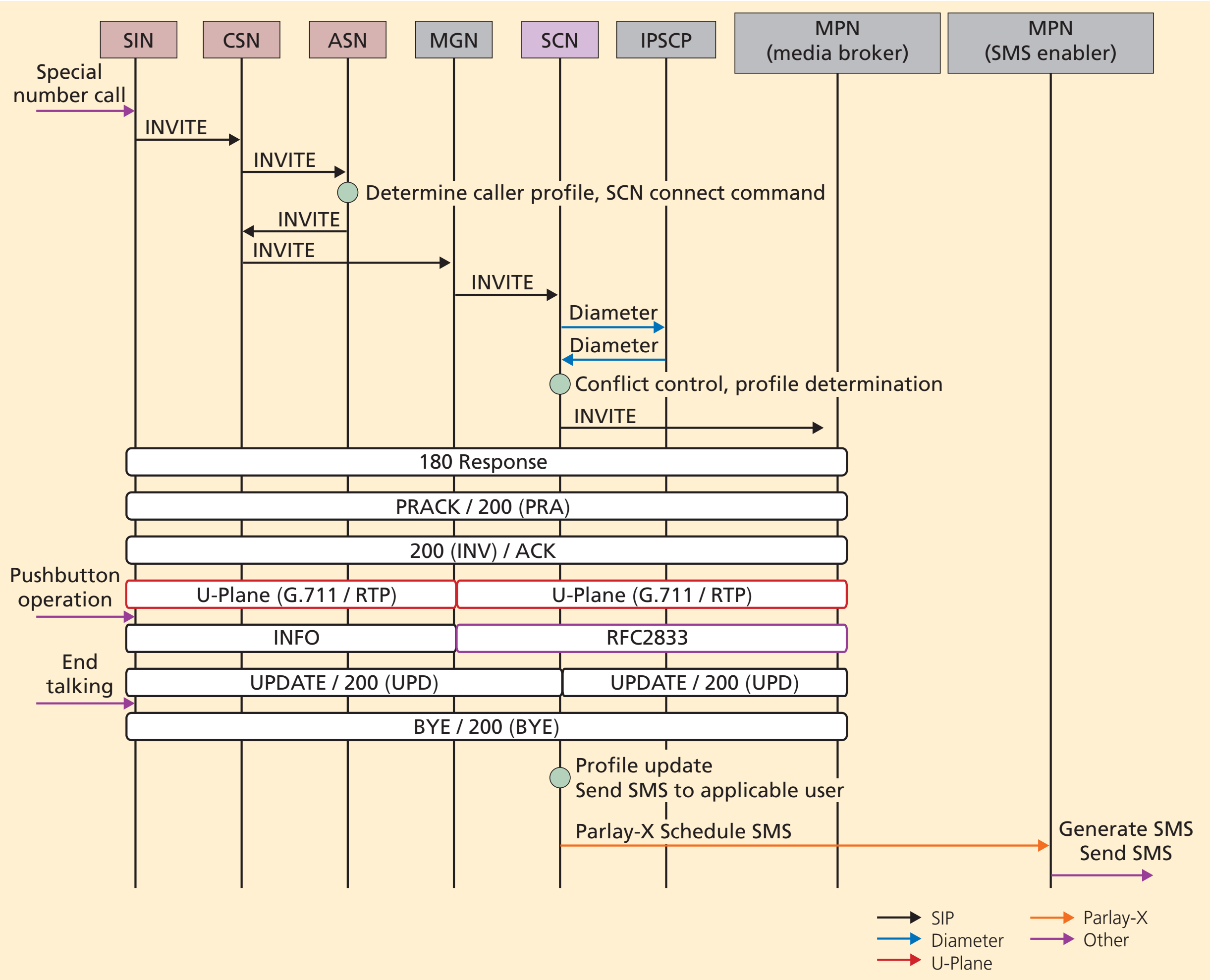

**Figure 5 Koe-no-Takuhaibin recording sequence**

*41 **ATM**: A communication scheme in which fixed-length frames called cells are transferred successively.

*42 **DTMF**: A method for transmitting telephone buttons and symbols using combinations of two out of four types of high and low pitched sounds.

*43 **RFC2833**: A specification for transmitting DTMF over RTP.

*44 **U-Plane**: A path for the transmission of user data to the C-Plane, which is a control signal transmission.

*45 **RTCP**: A communication protocol for exchanging data reception conditions from a streaming server and controlling the transmission rate. It is used in combination with Real-time Transport Protocol (RTP), which is a communication protocol for distributing audio and video streaming data in real time.

## 6. Conclusion

Through the introduction of SCN and use of abstract APIs provided by enablers, we are able to provide network services more quickly and more economically than ever before.

In the future, we will encourage creation of enablers for existing telecom functions, and introduction of new enablers to support a wider range of services. We will also continue to address the issue of realizing added value such as only an operator can provide.

As these new enablers and services increase and the network evolves, we will also study introduction of new functions such as access control for enablers, and control of conflicts between these new services.

REFERENCES


[1] ETSI ES 202 504-2: "Open Service Access (OSA); Parlay X Web Services; Part 2: Third Party Call (Parlay X 3)," Jun. 2007.

[2] 3GPP TS 24.229 V6.24.0: "IP multimedia call control protocol based on Session Initiation Protocol (SIP) and Session Description Protocol (SDP); Stage 3," Dec. 2009.

[3] 3GPP TS 29.328 V6.15.0: "IP Multimedia (IM) Subsystem Sh interface; Signalling flows and message contents," Sept. 2009.

[4] ETSI ES 202 504-4: "ETSI ES 202 504-4: Open Service Access (OSA); Parlay X Web Services; Part 4: Short Messaging (Parlay X 3)," May 2008.