WIRELESS COMMUNICATIONS AND MOBILE COMPUTING
Wirel. Commun. Mob. Comput. 2007; **7**:755–766
Published online 12 September 2006 in Wiley InterScience (www.interscience.wiley.com). DOI: 10.1002/wcm.406

Design and implementation of UMTS session management in the user equipment

Chai-Hien Gan[1*,†], Yi-Bing Lin[1] and Shi-Hi Chen[2]

[1]*Department of Computer Science, National Chiao Tung University, Hsinchu 300, Taiwan*
[2]*Networks and Multimedia Institute, Institute for Information Industry, Taipei 106, Taiwan*

Summary

In universal mobile telecommunications system (UMTS), session management (SM) maintains a communication session between a user equipment (UE) and the core network. 3GPP TS 24.008 specifies the SM functions and the communication protocols between the UE and the core network. However, the interaction between the SM and other entities in the UE are not specified in detail. This paper designs and implements the SM software. We use a finite state machine (FSM) to model the SM functions. Based on the FSM and the primitives, we describe how to set up the development steps to implement the SM at the UE. Copyright © 2006 John Wiley & Sons, Ltd.

KEY WORDS: universal mobile telecommunications system (UMTS); session management (SM); packet data protocol (PDP) context; packet-switched (PS) domain; user equipment (UE)

1. Introduction

Universal mobile telecommunications system (UMTS) [1] evolves from the general packet radio service (GPRS) to provide wideband data services. With UMTS, wireless multimedia communication can be offered with high quality, and access to services in the Internet will be enhanced by higher data rates and more flexible communication capabilities.

As shown in Figure 1, UMTS consists of the terrestrial radio access network (UTRAN; Figure 1 (1)) and the core network (Figure 1 (2)). The User Equipment (UE; Figure 1 (3)) communicates with the UTRAN through the radio access bearer (RAB; including the radio link). In UTRAN, the radio network controller (RNC) is responsible for control of node Bs (base stations) for RAB, and maintains the links to the core network (i.e., serving GPRS support node or SGSN) to provide the PS domain services.

Session management (SM) [2,3] maintains the routing path of a communication session between a UE and the core network. The SM provides packet routing functions including Internet protocol (IP) address assignment, quality of service (QoS) setting, and so on.

*Correspondence to: Chai-Hien Gan, Department of Computer Science, National Chiao Tung University, Hsinchu 300, Taiwan.
†E-mail: chgan@csie.nctu.edu.tw

Contract/grant sponsor: NSC Excellence project; contract/grant numbers: NSC 94-2752-E-009-005-PAE; NSC 94-2219-E-009-001; NSC 94-2213-E-009-104.
Contract/grant sponsor: NTP VoIP Project; contract/grant number: NSC 94-2219-E-009-002.
Contract/grant sponsor: NTP Service IOT Project; contract/grant number: NSC 94-2219-E-009-024.
Contract/grant sponsors: Intel; IIS/Academia Sinica; ITRI/NCTU Joint Research Center.

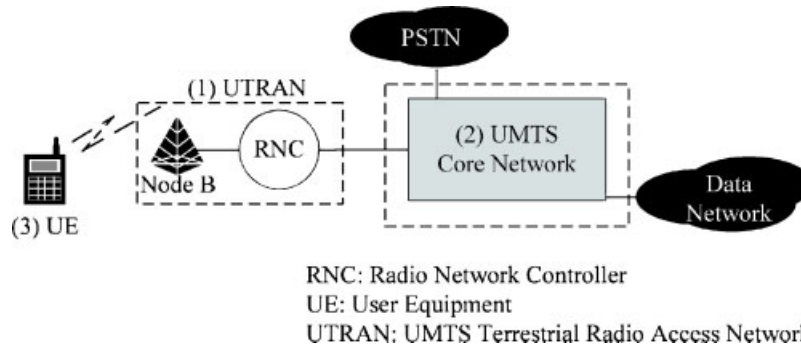

Fig. 1. The UMTS network architecture.

Before a UE accesses any PS domain service, a packet data protocol (PDP) context for the service must be activated to specify the routing information. The PDP context is maintained by the SM in both the UE and the core network. Figure 2 shows the UE protocol architecture for supporting SM functions, in which the service access points (SAPs) are marked with circles [4]. The access stratum (AS) sublayer (Figure 2 (1)) provides services to the mobility management (MM) sublayer (Figure 2 (2)) and the RAB manager (RABM) entity (Figure 2 (3)). The GPRS mobility management (GMM; Figure 2 (4)) of the MM sublayer provides services to the SM entity (Figure 2 (5)) of the connection management (CM; (Figure 2 (6)), and the ATtend (AT; Figure 2 (7)) entity provides services (i.e., AT commands) for the applications to invoke the SM functions.

The SM includes procedures for PDP context activation, deactivation, and modification [2]. Through the SMREG-SAP, the SM allows the AT to invoke the SM functions. Through the RABMSM-SAP, the SM informs the RABM about the PDP context information; that is, negotiated QoS profile and active network service access point identifier (NSAPI) for transmitting protocol data units (PDUs).

The GMM supports attach, detach, security, and location management functions. Through the GMMSM-SAP, the GMM allows the SM to send/receive Layer 3 protocol (L3) messages [2] to/from the core network (i.e., the SGSN). The SM procedures can only be performed if a GMM context has been established between the UE and the SGSN. Ongoing SM procedures are suspended during the GMM context establishment.

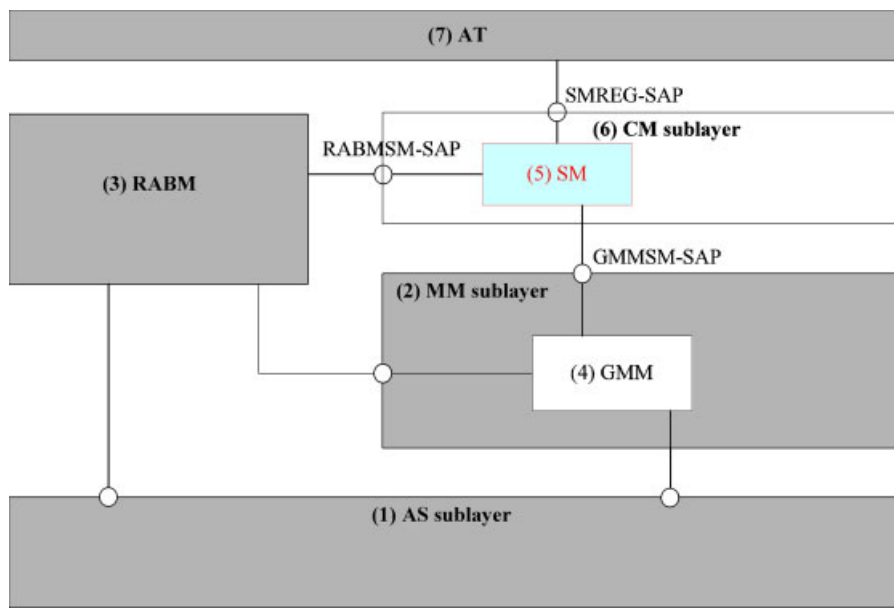

Fig. 2. The UE protocol architecture for SM.

In this paper, we propose a design and implementation of the SM software in the UE. We describe the SM functionality, and then elaborate on how the SM software can be developed from its finite state machine (FSM). The notation used in this paper is listed in the Appendix.

2. SM Functionality

This section describes UE SM functions. In UMTS, a UE can be engaged in multiple communication sessions at the same time. Every communication session at the UE is associated with an NSAPI value. The NSAPI values range from 5 to 15, where the maximal number of simultaneous sessions is 11 [2].

Our SM design follows the primitive flow model shown in Figure 3 [5]. This model includes two participants: the service user and the service provider. A primitive can be one of the following five types: REQ (Request), IND (Indication), RSP (Response), CNF (Confirm), and REJ (Reject). The REQ primitive is initiated by the service user of the initiator. The service provider of the initiator then delivers the request to the corresponding responder. When the service provider of the responder receives the request, it informs the service user by issuing the IND primitive. The service user of the responder then sends the RSP primitive to the service provider. After the service provider of the initiator receives this response, it issues the CNF/REJ primitive to the service user to confirm/reject the request.

Three SAPs, SMREG-SAP, GMMSM-SAP, and RABMSM-SAP interface the SM with the AT, the GMM, and the RABM, respectively. The interaction between the SM and an entity (e.g., the GMM) takes place when a primitive is sent across the corresponding SAP (e.g., the GMMSM-SAP). The standardized SAP primitives defined in 3GPP TS24.007 [4] allow independent implementation of the SM software to achieve the modularity goal. This approach also guarantees compatibility with future UMTS versions.

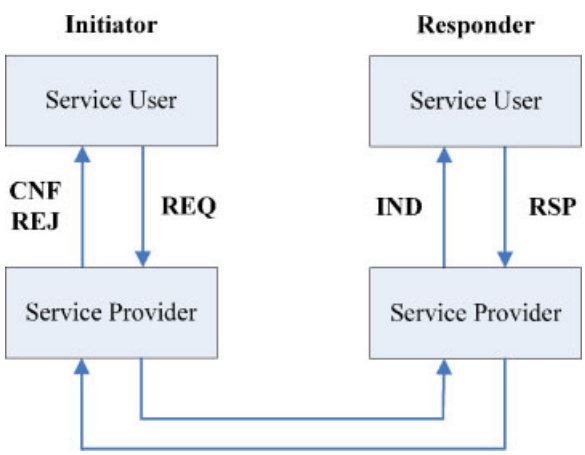

Fig. 3. Primitive flow model.

According to 3GPP TS24.007, every SM primitive issued from/to the AT or the RABM contains an NSAPI value. As we previously mentioned, the NSAPI value is used to identify a communication session at the UE. On the other hand, the SGSN uses a transaction identifier (TI) value to identify a communication session between the UE and the SGSN. Therefore, the primitive issued from/to the GMM (used to exchange the requests/responses between the UE and the SGSN) contains a TI value. To identify the NSAPI corresponding to a TI, the SM maintains a TI-to-NSAPI mapping table. By retrieving this mapping table, the SM determines the communication session specified by the GMM primitive. Primitives for SAPs are elaborated as follows.

2.1. Primitives for SMREG-SAP

The SMREG-SAP interfaces the SM with the AT. This SAP provides the primitives for PDP context procedures. For these primitives, the SM is the service provider, and the AT is the service user. The primitives are described as follows:

- SMREG-PDP-ACTIVATE: The REQ primitive initiates PDP context activation. The IND primitive is issued by the SM. This primitive instructs the AT to activate a PDP context requested from the SGSN. Then the AT checks whether it can initiate PDP context activation. If so, the AT issues the REQ primitive to initiate the activation task. Otherwise, the RSP primitive replies that the PDP context cannot be initiated by the AT. The CNF/REJ primitive indicates the result for the execution of PDP context activation procedure.
- SMREG-PDP-DEACTIVATE: The primitives implement the PDP context deactivation procedure.
- SMREG-PDP-MODIFY: The primitives modify a PDP context.
- SMREG-PDP-ACTIVATE-SEC: The primitives implement the secondary PDP context activation procedure.

2.2. Primitives for GMMSM-SAP

The GMMSM-SAP interfaces the SM with the GMM, where the SM is the service user, and the GMM is the

service provider. The primitive types can be REQ, CNF, REJ, or IND, and are described as follows.

- GMMSM-ESTABLISH: The REQ primitive initiates the establishment of a GMM context. The CNF/REJ primitive indicates that the GMM context establishment procedure has been concluded.
- GMMSM-RELEASE: The IND primitive informs the SM that the UE has been detached. The RSP primitive does not exist because the GMM context has been released at that time.
- GMMSM-UNITDATA: These primitives are used to deliver the L3 messages defined in 3GPP TS 24.008 [2]. The REQ primitive requests the GMM to forward the L3 message to the SGSN. Upon receipt of the L3 message from the SGSN, the GMM forwards it to the SM by using the IND primitive.

2.3. Primitives for RABMSM-SAP

In RABMSM-SAP, the SM is the service provider, and the RABM is the service user. The primitive types can be IND or RSP, and are described as follows.

- RABMSM-ACTIVATE: The IND primitive informs the RABM that an NSAPI has been activated for data transfer. It also informs the RABM about the QoS profile negotiated. The RSP primitive informs the SM that the indicated NSAPI is now in use, and the RAB is established.
- RABMSM-DEACTIVATE: The IND primitive requests the RABM to de-allocate an NSAPI. The RSP primitive informs the SM that the indicated NSAPI is no longer in use, and the RAB has been released.
- RABMSM-MODIFY: The IND primitive requests the RABM to modify the QoS profile for an NSAPI. The RSP primitive informs the SM that the indicated NSAPI and the QoS profile negotiated are now in use.

3. Implementation of the SM Software

This section uses a FSM tool called *development environment for protocol coding and testing* (DEPOT) to implement the SM software at the UE. We model the SM software as a FSM, and show how to follow the DEPOT development steps to implement the SM software from its FSM.

3.1. DEPOT

DEPOT is developed by the Institute for Information Industry (III) [6]. This tool defines several macros and functions in the C programming language [7] to assist development of the software from the FSM. The developed source codes are compiled together with DEPOT macros and functions. Figure 4 shows the DEPOT framework architecture. The framework (Figure 4 (1)) consists of a global queue (Figure 4 (2)), a global dispatcher (Figure 4 (3)), and several user specified

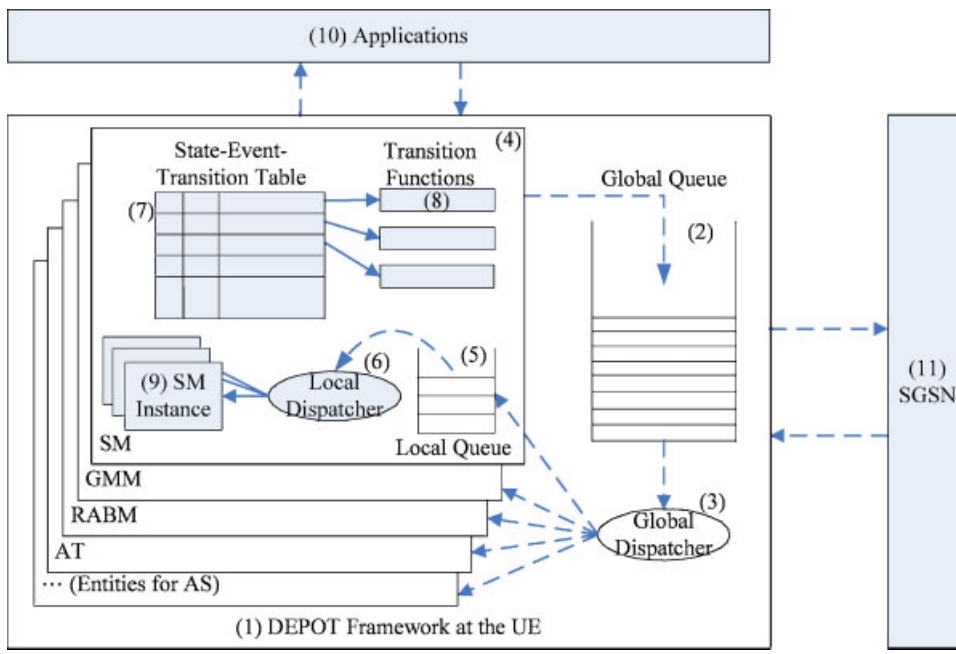

Fig. 4. The DEPOT framework architecture at the UE.

entities (Figure 4 (4); e.g., AT, RABM, SM, and GMM). Each entity contains a local queue (Figure 4 (5)), a local dispatcher (Figure 4 (6)), a state-event transition table (Figure 4 (7)), and several transition functions (Figure 4 (8)). The entity may create multiple instances to handle the simultaneous services. For example, an SM instance (Figure 4 (9)) is used to maintain the status of a communication session. In the UE protocol, the AT entity provides services for applications (Figure 4 (10)) to access the UE functionalities (e.g., PDP context activation), and the UE uses the entities for the AS sublayer to communicate with the SGSN (Figure 4 (11)).

In DEPOT, a primitive (e.g., SMREG-PDP-ACTIVATE-REQ) described in Section 2 is implemented as an event (e.g., the SMREG-PDP-ACTIVATE-REQ event) that triggers the execution of a transition function and possibly generates new events to other entities. Every event in DEPOT is associated with a *DEPOT message* that contains the parameters for transition function execution. For example, the DEPOT message associated with the SMREG-PDP-ACTIVATE-REQ event includes the PDP address, the access point name (APN) [8], the required QoS and so on, which are used to activate a PDP context.

All invoked events are first buffered in the global queue. The global dispatcher forwards every event to the local queue of the corresponding entity. In each entity (e.g., the SM entity), the local dispatcher retrieves the event from its local queue and identifies the instance (e.g., an SM instance) that should handle this event. Based on the FSM state of the instance, the retrieved event, and the state-event transition table, DEPOT determines the transition function to be executed at the instance. To use DEPOT, developers should define a state-event transition table, and implement the local dispatcher and the transition functions for each entity.

3.2. Developing the SM Software from its FSM

We use PDP context activation as an example to illustrate how to develop the SM software from its FSM. Figure 5 shows the execution flow of UE-initiated PDP context activation. In this figure, a solid arrow represents a primitive invoked by an instance of a UE entity. A dashed arrow represents an L3 message delivered between the UE and the SGSN. To perform this task, the SM maintains a FSM to handle the interaction with other entities (i.e., the AT, the RABM, and the GMM). In Figure 5, an SM FSM state is represented by an oval. Figure 6 illustrates an incomplete state transition diagram of the SM FSM. Initially, the FSM is in the INACTIVE state indicating that the PDP context has not been activated.

Step 1. The AT invokes the SMREG-PDP-ACTIVATE-REQ event that requests the SM to activate a PDP context (Figure 5 (1.1)). Upon receipt of the SMREG-PDP-ACTIVATE-REQ event, the SM invokes the RABMSM-ACTIVATE-IND event that informs the RABM about the NSAPI and the QoS profile regarding this PDP context activation (Figure 5 (1.2)), and invokes the GMMSM-UNITDATA-REQ event that requests the GMM to forward the L3 message ACTIVATE PDP CONTEXT REQUEST to the SGSN (Figure 5 (1.3)) [2]. Then the SM starts the T3380 timer (Figure 5 (1.4)). The FSM enters the ACTIVE-PENDING state (see Transition 1 in Figure 6). This state indicates that PDP context activation was requested by the AT. The GMM will forward the L3 message to the SGSN (Figure 5 (1.5)). We assume that this L3 message is lost, and will be retransmitted at Step 3.

Step 2. Suppose that the response (i.e., the RABMSM-ACTIVATE-RSP event) from the RABM arrives before T3380 expires (Figure 5 (2.1)). The FSM moves from ACTIVE-PENDING to ACTIVE-WAIT-GMM (see Transition 5 in Figure 6). In this state, the SM waits for the GMM response (e.g., if the SGSN accepts the activation for the PDP context).

Step 3. On expiry of the T3380 timer (Figure 5 (3.1)), the GMMSM-UNITDATA-REQ event is re-invoked by the SM (Figure 5 (3.2)). The FSM remains at ACTIVE-WAIT-GMM (see Transition 7 in Figure 6). Then the GMM retransmits the L3 message ACTIVATE PDP CONTEXT REQUEST to the SGSN again (Figure 5 (3.3)). We assume that this L3 message is received by the SGSN.

Step 4. Suppose that the SGSN replies the UE with the L3 message ACTIVATE PDP CONTEXT ACCEPT (Figure 5 (4.1)) [2]. Upon receipt of the L3 message, the GMM invokes the GMMSM-UNITDATA-IND event to inform the SM that the activation has been accepted by the SGSN (Figure 5 (4.2)). Then the SM stops the T3380 timer (Figure 5 (4.3)), and invokes the SMREG-PDP-ACTIVATE-CNF event to confirm the completion of the PDP context activation (Figure 5 (4.4)). The FSM moves from ACTIVE-WAIT-GMM to ACTIVE (see

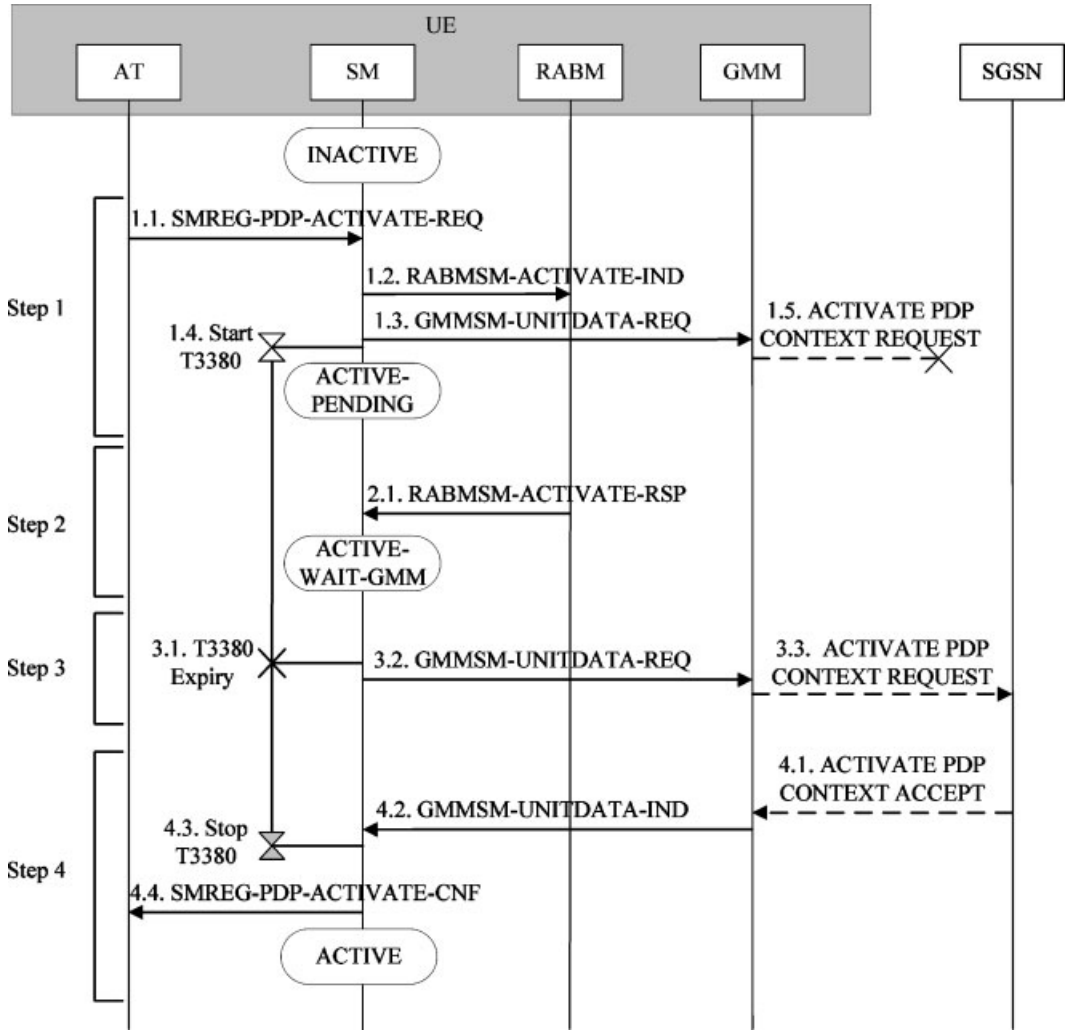

Fig. 5. UE-initiated PDP context activation.

Transition 9 in Figure 6). This state indicates that the PDP context is successfully activated.

As shown in Figure 5, when a PDP context activation is requested, the SM instance must wait for responses from both the GMM and the RABM. These responses may arrive in an arbitrary order. For example, when the GMM response arrives at the ACTIVE-PENDING state, the SM must check whether the RABM response has already arrived. To reduce the complexity of SM function execution, we introduce two states ACTIVE-WAIT-GMM and ACTIVE-WAIT-RABM to process the arbitrary arrival events. These two states are not defined in 3GPP specification [2,4]. They are introduced in our implementation to simplify event handling.

In Figure 6, we omit the details for PDP context modification and deactivation, which are similar to PDP context activation. In this figure, the events associated with a transition are represented by the form 'A/ B', where A is the input event and B represents one or more output events. If there is no output event for a transition, B is omitted. The transitions for PDP context activation (Figure 6) are not systematically elaborated in 3GPP specification, and are described as follows:

Figure 6 (1): SMREG-PDP-ACTIVATE-REQ/GMM-SM-UNITDATA-REQ, RABMSM-ACTIVATE-IND. This transition is described at Step 1 in Figure 5. The FSM moves from INACTIVE to ACTIVE-PENDING.

Figure 6 (2.1): GMMSM-UNITDATA-IND (*Reject*)/ RABMSM-DEACTIVATE-IND, SMREG-PDP-ACTIVATE-REJ. Upon receipt of the

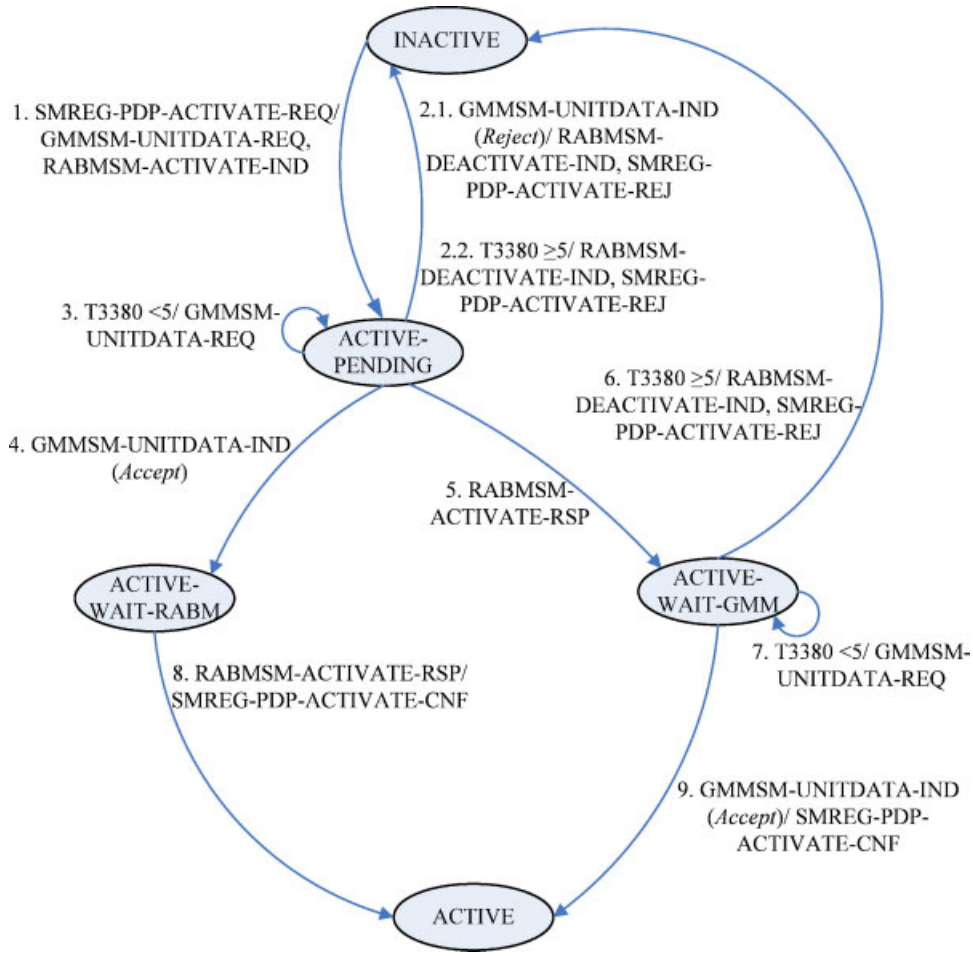

Fig. 6. An incomplete SM FSM state transition diagram (for PDP context activation only).

GMMSM-UNITDATA-IND event associated with the L3 message ACTIVATE PDP CONTEXT REJECT [2], the SM invokes the RABMSM-DEACTIVATE-IND event that requests the RABM to de-allocate an indicated NSAPI, and invokes the SMREG-PDP-ACTIVATE-REJ event to inform the AT that the PDP context activation is rejected by the SGSN. The FSM moves from ACTIVE-PENDING to INACTIVE.

Figure 6 (2.2): T3380 $\geq$ 5/ RABMSM-DEACTIVATE-IND,SMREG-PDP-ACTIVATE-REJ. On the fifth expiry of the T3380 timer, the SM handles the event similar to that for Transition 2.1. The FSM moves from ACTIVE-PENDING to INACTIVE.

Figure 6 (3): T3380 $<$ 5/ GMMSM-UNITDATA-REQ. The T3380 expiry event is described at Step 3 in Figure 5. The GMMSM-UNITDATA-REQ event associated with the L3 message ACTIVATE PDP CONTEXT REQUEST is re-invoked by the SM. This action is repeated for at most four times. The FSM remains at ACTIVE-PENDING.

Figure 6 (4): GMMSM-UNITDATA-IND (*Accept*). Upon receipt of the GMMSM-UNITDATA-IND event associated with the L3 message ACTIVATE PDP CONTEXT ACCEPT, the FSM moves from ACTIVE-PENDING to ACTIVE-WAIT-RABM.

Figure 6 (5): RABMSM-ACTIVATE-RSP. This transition is described at Step 2 in Figure 5. Upon receipt of the RABMSM-ACTIVATE-RSP event, the FSM moves from ACTIVE-PENDING to ACTIVE-WAIT-GMM.

Figure 6 (6): T3380 $\geq$ 5/ SMREG-PDP-ACTIVATE-REJ, RABMSM-DEACTIVATE-IND. This transition is similar to Transition 2.2. The FSM moves from ACTIVE-WAIT-GMM to INACTIVE.

Figure 6 (7): T3380 < 5/ GMMSM-UNITDATA-REQ. This transition is similar to Transition 3. The FSM remains at ACTIVE-WAIT-GMM.

Figure 6 (8): RABMSM-ACTIVATE-RSP/ SMREG-PDP-ACTIVATE-CNF. Upon receipt of the RABMSM-ACTIVATE-RSP event, the SM invokes the SMREG-PDP-ACTIVATE-CNF event. This event informs the AT of the completion for PDP context activation. The FSM moves from ACTIVE-WAIT-RABM to ACTIVE.

Figure 6 (9): GMMSM-UNITDATA-IND (*Accept*)/ SMREG-PDP-ACTIVATE-CNF. This transition is described at Step 4 in Figure 5. The PDP context activation is accepted by the SGSN. The FSM moves from ACTIVE-WAIT-GMM to ACTIVE.

3.3. DEPOT Implementation for PDP Context Activation

Based on the FSM state transition diagram in Figure 6, we describe the DEPOT setup steps that implement PDP context activation for UE SM. We first define the state-event transition table. Then we show how to implement the local dispatcher and the transition functions.

3.3.1. The state-event transition table

Figure 7 illustrates the DEPOT definition for the state-event transition table, which are described as follows:

- Line 1 declares the SM entity *sm* by using the DEPOT macro DELCLARE_ENTITY.

```
1.  DECLARE_ENTITY (sm)

2.  DEFINE_STATE_START (sm)
3.      STATE (INACTIVE)
4.      STATE (ACTIVE-PENDING)
5.      STATE (ACTIVE-WAIT-GMM)
6.      STATE (ACTIVE-WAIT-RABM)
7.      STATE (ACTIVE)
8.      ...
9.  DEFINE_STATE_END (sm)

10. DEFINE_EVENT_START (sm)
11.     EVENT (SMREG-PDP-ACTIVATE-REQ)
12.     EVENT (SMREG-PDP-ACTIVATE-CNF)
13.     EVENT (SMREG-PDP-ACTIVATE-REJ)
14.     EVENT (T3380)
15.     ...
16. DEFINE_EVENT_END (sm)

17. DECLARE_TRANSITION_FN (inactiveSMREG-PDP-ACTIVATE-REQ)
18. DECLARE_TRANSITION_FN (active-pendingRABMSM-ACTIVATE-RSP)
19. DECLARE_TRANSITION_FN (active-wait-gmmT3380)
20. DECLARE_TRANSITION_FN (active-wait-gmmGMMSM-UNITDATA-IND)
21. ...

22. DEFINE_TRANSITION_TABLE_START (sm)
23.     TRANSITION  (INACTIVE, SMREG-PDP-ACTIVATE-REQ,
                inactiveSMREG-PDP-ACTIVATE-REQ)
24.     TRANSITION (ACTIVE-PENDING, T3380,
                active-pendingT3380)
25.     TRANSITION (ACTIVE-PENDING, GMMSM-UNITDATA-IND,
                active-pendingGMMSM-UNITDATA-IND)
26.     ...
27. DEFINE_TRANSITION_TABLE_END (sm)
```

Fig. 7. The definition of the state-event transition table for UE SM.

- Lines 2–9 define the FSM states for *sm* by using the macros DEFINE_STATE_START and DEFINE_STATE_END. Each state (e.g., ACTIVE, ACTIVE-PENDING, etc in Figure 6) is defined by using the STATE macro.
- Lines 10–16 define the events using the macros DEFINE_EVENT_START and DEFINE_EVENT_END. Each event (e.g., SMREG-PDP-ACTIVATE-REQ described in Subsection 3.2) is specified by the EVENT macro.
- Lines 17–21 declare the transition functions by using the DECLARE_TRANSITION_FN macro. Details for transition functions (e.g., inactiveSMREG-PDP-ACTIVATE-REQ) will be elaborated in Subsubsection 3.3.3.
- Lines 22–27 define the state-event transition table by using the macros DEFINE_TRANSITION_TABLE_START and DEFINE_TRANSITION_TABLE_END. Each entry in the table is defined by the TRANSITION macro consisting of three elements: the state *S*, the input event *E*, and the transition function *F*. The transition function *F* (e.g., inactiveSMREG-PDP-ACTIVATE-REQ) is performed when the input event *E* (e.g., SMREG-PDP-ACTIVATE-REQ) is received by an instance (e.g., SM instance) at the state *S* (e.g., INACTIVE).

3.3.2. The local dispatcher

This subsection describes the implementation of the local dispatcher. The event issued to the SM entity is first extracted by the local dispatcher. For an event issued from the AT or the RABM, the local dispatcher identifies the target SM instance according to the NSAPI value obtained from the DEPOT message. For an event issued from the GMM, the NSAPI of the SM instance is derived from the TI value. If the corresponding SM instance has not been created, it is immediately created by the local dispatcher. If no GMM context has been established when an event (e.g., SMREG-PDP-ACTIVATE-REQ) is received, the local dispatcher suspends the SM procedure and establishes a GMM context by using the GMM procedure [2,3]. After GMM context establishment, the suspended SM procedure is resumed. Based on Figure 5, we illustrate how the local dispatcher handles the events.

- As shown at Step 1 in Figure 5, the SMREG-PDP-ACTIVATE-REQ event is issued to the SM for requesting PDP context activation. The local dispatcher creates an SM instance by the DEPOT function SmCreateInstance (*sm*, *InstId*) where *sm* is the identifier of the SM entity, and *InstId* uniquely identifies the SM instance. In our design, *InstId* is set to the NSAPI value indicated by this event. The dispatcher will also choose an unused TI value (ranging from 0 to 127), and insert this TI and NSAPI pair to the TI-to-NSAPI mapping table. According to the defined state-event transition table (see Line 23 in Figure 7), this event is processed by executing the transition function inactiveSMREG-PDP-ACTIVATE-REQ (see Subsubsection 3.3.3 for the details). The FSM moves from INACTIVE to ACTIVE-PENDING.
- When the RABMSM-ACTIVATE-RSP event at Step 2 in Figure 5 arrives, the local dispatcher identifies the SM instance according to the NSAPI value indicated by this event. The transition function active-pendingRABMSM-ACTIVATE-RSP (declared at Line 18, Figure 7) is executed. The FSM moves from ACTIVE-PENDING to ACTIVE-WAIT-GMM.
- For the T3380 event at Step 3 in Figure 5 arrives, the associated DEPOT message includes an NSAPI value and an expiry counter. The expiry counter is used to count the number of expiry events invoked by this SM instance. If the expiry counter value is less than 5, the event is processed by executing the transition function active-wait-gmmT3380 (declared at Line 19, Figure 7). The FSM remains at ACTIVE-WAIT-GMM.
- The GMMSM-UNITDATA-REQ event at Step 4 in Figure 5 is associated with an L3 message ACTIVATE PDP CONTEXT ACCEPT. This L3 message contains a TI value, which is used to retrieve the NSAPI value from the TI-to-NSAPI mapping table. The SM instance identified by the NSAPI value invokes the transition function active-wait-gmmGMMSM-UNITDATA-REQ (declared at Line 20, Figure 7). The FSM moves from ACTIVE-WAIT-GMM to ACTIVE.

3.3.3. The transition function

Figure 8 shows the C code for the transition function inactiveSMREG-PDP-ACTIVATE-REQ. This function is performed when the SM receives the SMREG-PDP-ACTIVATE-REQ event at the INACTIVE state.

- Line 1 invokes the IMPLEMENT_TRANSITION_FN_START macro to start the transition function inactiveSMREG-PDP-ACTIVATE-REQ declared at Line 17 in Figure 7.
- Lines 2–4 declare three variables: pSMREG-PDP-ACTIVATE-REQ, pRABMSM-ACTIVATE-IND,

```
1. IMPLEMENT_TRANSITION_FN_START (inactiveSMREG-PDP-ACTIVATE-REQ)

2.   tSMREG-PDP-ACTIVATE-REQ *pSMREG-PDP-ACTIVATE-REQ;
3.   tRABMSM-ACTIVATE-IND *pRABMSM-ACTIVATE-IND;
4.   tGMMSM-UNITDATA-REQ *pGMMSM-UNITDATA-REQ;

5.   pSMREG-PDP-ACTIVATE-REQ= (tSMREG-PDP-ACTIVATE-REQ*) MESSAGE;

6.   pRABMSM-ACTIVATE-IND = DpAllocMessage (rabm, sm, RABMSM-ACTIVATE-IND,
       sizeof (tRABMSM-ACTIVATE-IND));
7.   FillRabmSmActMsg (pRABMSM-ACTIVATE-IND, pSMREG-PDP-ACTIVATE-REQ);

8.   pGMMSM-UNITDATA-REQ = DpAllocMessage (gmm, sm, GMMSM-UNITDATA-REQ,
       sizeof (tGMMSM-UNITDATA-REQ));
9.   FillGmmSmActMsg (pGMMSM-UNITDATA-REQ, pSMREG-PDP-ACTIVATE-REQ);

10.  DpSendMsg (pRABMSM-ACTIVATE-IND);
11.  DpSendMsg (pGMMSM-UNITDATA-REQ);
12.  DpStartTimer (T3380Timer [NSAPI]);

13.  CHANGE_STATE_TO (ACTIVE-PENDING);

14.  DpFreeMessage (MESSAGE);
15.  RETURN_TRANSITION_RESULT (TRANSITION-SUCCESS);

16. IMPLEMENT_TRANSITION_FN_END (inactiveSMREG-PDP-ACTIVATE-REQ)
```

Fig. 8. The C code of the transition function inactiveSMREG-PDP-ACTIVATE-REQ.

and pGMMSM-UNITDATA-REQ. These variables are used to store the DEPOT messages associated with the events SMREG-PDP-ACTIVATE-REQ, RABMSM-ACTIVATE-IND, and GMMSM-UNITDATA-REQ, respectively.

- Line 5 uses the MESSAGE macro to save the DEPOT message associated with the input event SMREG-PDP-ACTIVATE-REQ in the variable pSMREG-PDP-ACTIVATE-REQ. This message includes the PDP address, the APN, the requested QoS etc.
- Line 6 uses the DpAllocMessage function to allocate memory storage for the pRABMSM-ACTIVATE-IND variable, and to specify the sender (i.e., the SM entity *sm*), the receiver (i.e., the RABM entity *rabm*), and the invoked event (i.e., RABMSM-ACTIVATE-IND). Note that the RABMSM-ACTIVATE-IND event is issued from *sm* to *rabm* by invoking the DpSendMsg function to be described at Line 10, Figure 8.
- Line 7 uses the FillRabmSmActMsg function to fill the DEPOT message associated with the output event RABMSM-ACTIVATE-IND, and store it in pRABMSM-ACTIVATE-IND. This message includes the NSAPI value and the QoS class. In UMTS, there are four QoS class: conversational, streaming, interactive, and background classes [5]. In the DEPOT message, the QoS class is obtained from the requested QoS (i.e., a parameter of pSMREG-PDP-ACTIVATE-REQ at Line 5), and the NSAPI value is obtained by the DEPOT macro CURRENT_INSTANCE_ID.
- Line 8 is similar to Line 6, where *gmm* is the identifier of the GMM entity and the invoked event is GMMSM-UNITDATA-REQ.
- Line 9 uses the FillGmmSmActMsg function to fill the DEPOT message associated with the output event GMMSM-UNITDATA-REQ, and store it in pGMMSM-UNITDATA-REQ. This DEPOT message includes an L3 message PDP CONTEXT ACTIVATE REQUEST. In this L3 message, the TI value is retrieved from the TI-to-NSAPI mapping table. Other parameters (e.g., the PDP address, the requested QoS, and the APN) are obtained from pSMREG-PDP-ACTIVATE-REQ.
- Lines 10 and 11 use the DpSendMsg function to issue the RABMSM-ACTIVATE-IND and the GMMSM-UNITDATA-REQ events associated with the DEPOT messages stored in the variables pGMMSM-UNITDATA-REQ and pRABMSM-ACTIVATE-IND, respectively. These events are targeted at the RABM and the GMM.
- Line 12 starts the T3380 timer and expects to receive the response from the SGSN before the timer expires.

The corresponding SM instance is indexed by the NSAPI.
- Line 13 uses the CHANGE_TO_STATE macro to move the FSM to the ACTIVE-PENDING state.
- Lines 14–16 deallocate the memory storage for the associated DEPOT message of the input event, and then exit the transition function.

4. Concluding Remarks

This paper designed and implemented the UMTS session management (SM) software for UE. We modeled the SM as a FSM and described how to use a tool called DEPOT to implement the SM software. With the DEPOT development steps, we showed that it is transparent to convert the FSM design into the SM program in the C language. In addition, we follow the standardized SM primitives defined in 3GPP TS 24.007 to allow independent implementation of the SM to achieve the modularity goal and guarantee compatibility with future UMTS versions. The developed SM software for UE in this paper has been transferred to Via Technologies, Inc. and Chung Shan Institute of Science and Technology (CSIST), Taiwan. The advantages of our approach are summarized as follows.
Formal description:

1. FSM is a well-known tool for modeling software functions formally.
2. Based on the formal description, we can easily verify the accuracy and integrity of the software functions.

Implementation complexity: The DEPOT module design allows code reuse, and developers can significantly reduce the coding time. *Consistency between design and implementation:* The design and implementation are based on the states and transitions, which provides better consistency between design and implementation.
Integration with other entities:

1. By using the primitive flow model, each entity can use the standardized SAP primitives to integrate with other entities.
2. The standardized SAP primitives allow independent implementation of the SM software to achieve the modularity goal.

Acknowledgement

This work was sponsored in part by NSC Excellence project NSC 94-2752-E-009-005-PAE, NSC 94-2219-E-009-001, NSC 94-2213-E-009-104, NTP VoIP Project under grant number NSC 94-2219-E-009-002, NTP Service IOT Project under grant number NSC 94-2219-E-009-024, Intel, IIS/Academia Sinica, III, MOE ATU Program, and ITRI/NCTU Joint Research Center.

Appendix: Notation

APN	Access point name
AS	access stratum
AT	Attend
CM	Connection management
CNF	Confirm
CS	Circuit-switched
DEPOT	Development environment for protocol coding and testing
FSM	Finite state machine
GMM	GPRS mobility management
GPRS	General packet radio service
III	Institute for information industry
IND	Indication
IP	Internet protocol
L3	Layer 3 protocol
MM	Mobility management
NSAPI	Network service access point identifier
PDP	Packet data protocol
PDU	Protocol data unit
PS	Packet-switched
QoS	Quality of service
RAB	Radio access bearer
RABM	Radio access bearer management
REJ	Reject
REQ	Request
RNC	Radio network controller
RSP	Response
SAP	Service access point
SGSN	Serving GPRS support node
SM	Session management
TI	Transaction identifier
UE	User equipment
UMTS	Universal mobile telecommunications system
UTRAN	UMTS terrestrial radio access network

References

1. Bannister J, Mather P, Coope S. *Convergence Technologies for 3G Networks: IP, UMTS, EGPRS and ATM*. John Wiley & Sons: USA, 2004.
2. 3GPP. 3rd Generation Partnership Project; Technical Specification Group core Network; Mobile radio interface signalling layer 3; Stage 3. Technical Report Technical Specification 3G TS 24.008 version 4.14.0 (2004-06), 2004.
3. Pang A-C, Chen J-C, Chen Y-K, Agrawal P. Mobility and session management: UMTS vs. cdma2000. *IEEE Wireless Communications* 2004; **1**(4): 30–44.
4. 3GPP. 3rd Generation Partnership Project; Technical Specification Group core Network; Mobile radio interface signalling layer 3; General aspects . Technical Report Technical Specification 3G TS 24.007 version 4.4.0 (2005-01), 2005.
5. Lin Y-B, Chlamtac I. *Wireless and Mobile Network Architectures*. John Wiley & Sons: USA, 2001.

6. NMI, III. Interface Design Description for DEPOT Part 002; SME Manual. Technical Report 2.0.0, Institute for Information Industry, 2004.
7. Kernighan B-W, Ritchie D-M. *The C Programming Language*, 2nd edn. Prentice Hall, Inc.: USA, 1988.
8. Chen Y-K, Lin Y-B. IP connectivity for gateway GPRS support node. *IEEE Wireless Communications* 2005; **1**(12): 37–46.

Authors' Biograpies

Chai-Hien Gan was born in Malaysia in 1971. He received his B.S. degree in computer science from Tamkang University in 1994, Taipei County, Taiwan, and both his M.S. degree and Ph.D. in computer science and information engineering from National Taiwan University, Taipei, Taiwan, in 1996 and 2005, respectively. Since March 2005, he has been a research assistant professor in Department of Computer Science, National Chiao Tung University, Taiwan. His current research interests include wireless and mobile computing, personal communications services, and wireless Internet.

Yi-Bing Lin is chair professor and vice president of Research and Development, National Chiao Tung University. His current research interests include wireless communications and mobile computing. Dr. Lin has published over 200 journal articles and more than 200 conference papers. Lin is the co-author of the books *Wireless and Mobile Network Architecture* (with Imrich Chlamtac; published by Wiley, 2001) and *Wireless and Mobile All-IP Networks* (with Ai-Chun Pang; published by Wiley, 2005). Lin is an IEEE fellow, ACM fellow, AAAS fellow, and IEE fellow.

Shi-Hi Chen is currently the director of the Wireless Network Technology Center of Network and Multimedia Institute of III. He received his Ph.D. in electrical engineering from University of Maryland at College Park in 1996. In 1997, he joined Motorola corporation as a member of technical staff, where he conduct wireless transmission simulation and system design. After leaving Mororola, he entered III to start 3G technology development in 1999. In III, he has been serving as the project manager or co-manager for sereval DoIT technolgy development projects, all on wireless communication technologies. Those projects have successfully generated several IP's and product reference design for transfer to local industry on WLAN AP, Bluetooth Stack, UWB, and 3G. He received a best DoIT project award on 2000.