Mobile Cloud Computing System Components Composition Formal Verification Method Based on Space-Time Pi-Calculus

Peng Wang[1,2](✉), Ling Yang[2], and Guo Wen Li[3]

[1] School of Management Fudan University, Shanghai, China
wangpeng@shcloudvalley.com
[2] Shanghai Cloud Valley Development Co., Ltd., Shanghai, China
yangling@shcloudvalley.com
[3] Shanghai Yangpu Science and Technology Innovation Group Co., Ltd., Shanghai, China
liguowen@shcloudvalley.com

Abstract. To build different mobile cloud computing system (MCS) business applications, how to design open system architecture is essential. First, a service-oriented architecture is put forward. In this architecture, MCS components are expressed as the form of interoperable MCS services, which are combined to achieve complex business needs. Second, a formal method is proposed based on space-time (S-T) Pi-calculus, in order to verify validity of MCS service composition model. Finally, the case study shows that how to apply the model and method. The experiment result shows that they are feasible.

Keywords: Mobile cloud computing · Service composition · Service-oriented architecture · Pi-calculus

1 Introduction

Mobile Cloud Computing is the combination of cloud computing, mobile computing and wireless networks to bring rich computational resources to mobile users, network operators, as well as cloud computing providers [1]. As the evolution of cloud computing, mobile cloud computing network resources are virtualized and allocated a set number of distributed computers rather than in the traditional local computer or servers. And they are assigned to the mobile devices such as intelligent mobile phone, smart terminal, etc. [2, 3]. On account of mobile cloud computing system (MCS) includes physical device, it is complex to design the architecture of MCS. Sat. M. proposed a system architecture, which uses virtual platform to offer personalized services to mobile device, but it doesn't resolve latent WAN latency issues [4]. Zhang X. put forward a flexible application programming model which constructed by CloneCloud, forever it's difficult to provide middleware to storage data mechanisms at terminal and cloud side [5].

This work is supported by China Postdoctoral Science Foundation funded project (No. 2014M 551334), special fund for Shanghai Zhangjiang National Innovation Demonstration Zone (No. ZJ2015ZD006).

In this paper, we proposed a novel SOA [6] architecture, in which hardware and software of MCS are integrated together in the form of compatible services. And these services can be combined to implement complex operational requirements.

Formal validation method is indispensable to verify the stability MCS service composition model, and the researches on it have achieved the rapid development, such as Finite State Machines [7] and Petri Net [8]. However, state-space-explosion of using charts is brought a big challenge. Pi-calculus [9] is able to represent concurrent computations as a formal analysis tool, although it's bound by space and time constraints, which is included in MCS [10]. In this paper, S-T Pi-calculus is presented by bring space and time operator into Pi-calculus, and a formal verification for MCS components composition is proposed based upon.

The remainder of this paper is organized as follows. The next section presents the conceptions of MCS service and S-T Pi-calculus. Section 3 devises a formal verification for verifying validity of MCS service composition. Section 4 presents a case study to show the applicability of model and formal verification. The paper concludes in Sect. 5.

2 Basic Conceptions

2.1 MCS Service View

We regard MCS as a combination of encapsulated MCS services, which includes units of mobile terminal, cloud computing and communications, as shown in Fig. 1. Mobile terminal unit is composed of computing terminals, sensors, and actuators. Communications unit offers communicative mechanism by utilizing mobile communication network. Cloud computing unit realizes functions of computation, control and storage, and it implements temporal and spatial management. Discrete domain blends with continuous domain in this unit.

MCS service interfaces' implementation details are hidden, which can be realized by different technologies from any service provider. There are characteristics, operation guide, access protocol, parameters and data types in interfaces description. The interfaces are provided to receipt and transmit messages, and it would take time to finish the receipt-transmit acts. As described above, we define MCS service view as follows.

Definition 1 (MCS Service View). $\equiv MCSV{=}(R, r_0, S, U, W, E, f, g)$. A brief description of each element is listed below.

$R = \{r_0, r_1, \ldots\}$: finite states set of MCS service.

r_0: initial state.

S: final states set, $S \subseteq R$.

$U = Act \cup \overline{Act} \cup \{\tau\}$: act set. $Act = \{a | a \in \Sigma\}$ is receipting act set (Σ is alphabet); $\overline{Act} = \{\overline{a} | a \in \Sigma\}$ is transmitting act set; τ is internal act.

$W \subseteq R \times U \times R$: relation of state transformation.

E: MCS service messages set.

f:$U \to E$ act-message function. $\forall x \in U$, $f(x)$ represent receipting or transmitting messages of x.

g:$U \to R^+$ act-time function. $\forall x \in U$, $g(x)$ represent the time spending in finishing x.

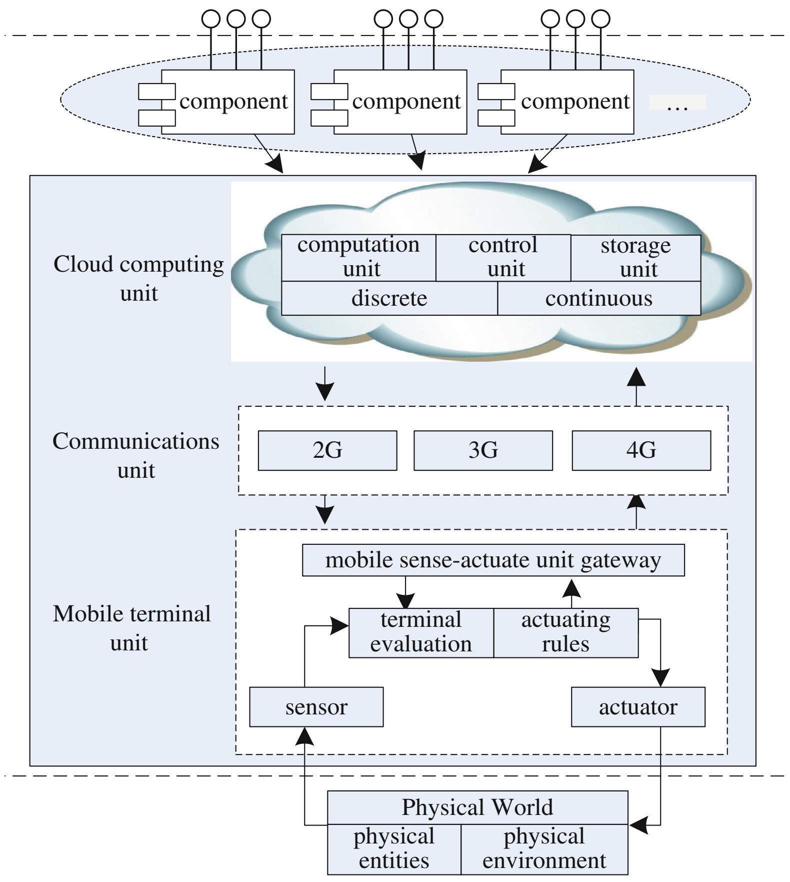

Fig. 1. MCS service structure block diagram

2.2 S-T Pi-Calculus

By definition, there is a tight correspondence between process of Pi-calculus and MCS service. Concretely speaking, channel corresponds to act; transmitting and receipting variable corresponds to transmitting and receipting message of act; composition, process, summation, replication in Pi-calculus correspond to structures of parallel, sequence, case and iterative. Nevertheless, there are no syntaxes representing temporal and spatial features. In this paper, we propose space and time operators, and define what syntaxes and operational semantics of S-T Pi-calculus are.

Physical modules of MCS are abstracted to spatial objects based on topological relation theory of spatial database [11]. Topological relations between two spatial objects, which are regarded as point sets, are expressed by a quaternion formed by boundary and interior of point set. Let M and N indicate two geospatial objects, and let $\partial M, M^0, \partial N, N^0$ represent interior and boundary. The quaternion is $\mathbf{R}(M,N) = \begin{pmatrix} \partial M \cap \partial N & \partial M \cap N^0 \\ M^0 \cap \partial N & M^0 \cap N^0 \end{pmatrix}$. There are eight kinds of topology models i.e., inside, disjoint, contain, meet, overlap, equal, covered, and cover by, and they are indicated by $S_{loc} = \{s_{in}, s_{ct}, s_d, s_m, s_o, s_e, s_c, s_{cb}\}$. Supposing Q includes m physical modules, and let d_i represents relationship between benchmark region and the i-th module's location. $\exists\ S_i \subseteq S_{loc}$, this module could work properly meeting criteria of $d_i \in S_i$. Let $S = \{S_1, S_2, \ldots, S_m\}$. Define space operator below.

Definition 2 (Space Operator). $\equiv Loc[S]$. $Loc[S]Q$ expresses that Q can start only when $\prod_{i=1}^{n} d_i \in S_i$ is true.

In this paper, discrete time domain is adopted to describe time characteristic of MCS. Properties of discrete time domain are defined as follows.

Definition 3 (Properties of Discrete Time Domain). Discrete time domain T has following properties.

(1) $\forall t \in T,\ t \neq \infty \Rightarrow \infty > t$;
(2) $\forall t \in T,\ t \neq 0 \Rightarrow t > 0$;
(3) $\forall t \in T,\ t + 0 = t,\ t + \infty = \infty$;
(4) $\forall t, t' \in T,\ t > t' \Leftrightarrow \exists \Delta t > 0,\ t' + \Delta t = t$;
(5) $\forall t, t' \in T,\ \ (t > 0) \wedge (t' \neq \infty) \Rightarrow t' + t > t'$;
(6) $\forall t_2, t_3 \in T,\ \forall [t_1,\ t_4],\ \exists t', t' \in T,\ t_2 \leq t' \leq t_4$, then $t' \in [t_1,\ t_2]$;
(7) $\forall t_1, t_2 \in T,\ t_1 > t_2,\ \ \{t | t_1 \leq t \leq t_2\}$ is represented as $[t_1,\ t_2]$.

Definition 4 (Time Operator). $\equiv Int(t_r, \Delta t)$, t_r is reference time, $\Delta t \geq 0$. $Int(t_r, \Delta t)Q$ represents Q can start only when it meets $t \in [t_r,\ t_r + \Delta t]$.

Definition 5 (Syntax of S-T Pi-Calculus).

$$\begin{aligned} Q ::= & 0|\overline{a}\langle x\rangle.Q|a(x).Q|\tau.Q|Q+P|Q|P|(x)Q|[x=y]Q|!Q| \\ & Loc[S]Q|Int(t_r,\Delta t)Q \end{aligned} \tag{1}$$

See reference [10] for concrete meanings of the above expressions.

Definition 6 (Operational Semantics of S-T Pi-Calculus).

(1) Space operator.

$$\frac{Q \xrightarrow{\alpha} Q'}{Loc[S]Q \xrightarrow{\alpha} Q'},\ c_i \in S_i;$$
$$\frac{Q \xrightarrow{\alpha} Q'}{Loc[S]Q \xrightarrow{\alpha} 0},\ c_i \notin S_i;$$

(2) Time operator ($\alpha \in \{\tau,\ a(x),\ \overline{a}\langle x\rangle\}$).

$$\frac{Q \xrightarrow{\alpha} Q'}{Int(t_r, \Delta t)Q \xrightarrow{\alpha} Q'}, \ t \in [t_r, t_r + \Delta t];$$

$$\frac{Q \xrightarrow{\alpha} Q'}{Int(t_r, \Delta t)Q \xrightarrow{\alpha} 0}, \ t \notin [t_r, t_r + \Delta t];$$

See reference [10] for other operational semantics.

3 A Formal Verification for MCS Components Composition

Basic MCS service portfolio modes include sequence structure, selection structure, parallel structure and recursive structure. They can be expressed as S-T Pi–calculus process expressions as follows.

(1) Sequence structure

$MCSV_1$ sends a message msg to $MCSV_2$ from port a, and then $MCSV_2$ receives this message from port b, as shown in Fig. 2.

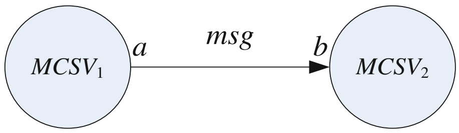

Fig. 2. Sequence structure

This structure can be expressed as following process expressions.

$$\bar{a}\langle msg\rangle.MCSV_1 \text{ and } b(msg).MCSV_2$$

(2) Selection structure

$MCSV_1$ selects to send one message from port a. If it sends msg_1, $MCSV_2$ would receive this message from port b. If it sends msg_2, $MCSV_3$ would receive this message from port c, as shown in Fig. 3.

This structure can be expressed as following process expressions.

$$(\bar{a}\langle msg_1\rangle.MCSV_1) + (\bar{a}\langle msg_2\rangle.MCSV_1),$$
$$b(msg_1).MCSV_2 \text{ and } c(msg_2).MCSV_3$$

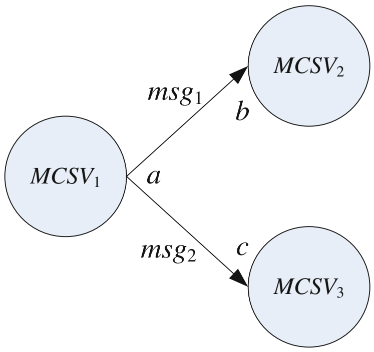

Fig. 3. Selection structure

(3) Parallel structure

$MCSV_1$ sends a message msg_1 to $MCSV_2$ from port a, and then $MCSV_2$ receives this message from port c. Meanwhile, $MCSV_1$ also sends a message msg_2 to $MCSV_2$ from port b, and then $MCSV_2$ receives this message from port d, as shown in Fig. 4.

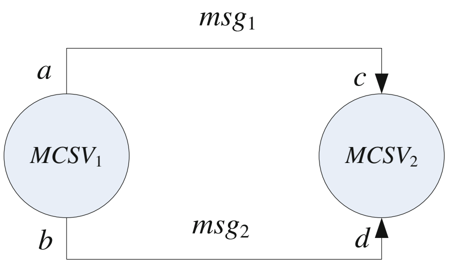

Fig. 4. Parallel structure

This structure can be expressed as following process expressions.

$(\bar{a}\langle msg_1\rangle.MCSV_1)|(\bar{b}\langle msg_2\rangle.MCSV_1)$ and $(c(msg_1).MCSV_2)|(d(msg_2).MCSV_2)$

(4) Recursive structure

$MCSV_1$ sends the same message msg to $MCSV_2$ repeatedly from port a, and $MCSV_2$ receives this message constantly from port b, as shown in Fig. 5.

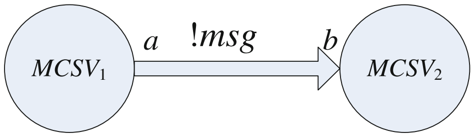

Fig. 5. Recursive structure

This structure can be expressed as following process expressions.

$$!(\bar{a}\langle msg\rangle.MCSV_1) \text{ and } !(b(msg).MCSV_2)$$

A formal verification is put forward for MCS service composition as the following three steps. First, a S-T Pi–calculus model is constructed based on MCS service portfolio modes. Secondly, temporal and spatial attributes of actual MCS services are introduced into the Pi–calculus model. Thirdly, from the process expression being deadlock or not, we can judge whether this MCS service composition can reach the final state. Then, the validity of MCS service composition can be verified.

4 Experiments and Results

In this section, We take Parking Electric Fence in Shanghai YangPu center business circle for example, to demonstrate the formal verification about building and validating a MCS components composition. Business case is described below. If a owner wants to stop, his request from intelligent terminal will be sent to vehicle dispatch center. In case there are no spaces available, the dispatch center would set electric fence. Meanwhile all cars inside the virtual fence would receive early warnings periodically.

Parking electric fence is designed to four MCS services as shown in Fig. 6. Let $P_{pr}, P_{vs}, P_{vf}, P_{ew}$ denote Parking Request, Vehicle Scheduling, Virtual Fence, and Early Warning, respectively. We make virtual fence benchmark region. Early Warning is inside i.e., $S_{ew} = \{\{s_{in}\}, \{s_{in}\}, \ldots, \{s_{in}\}\}$. The other ones are unconstrained i.e., $S_{pr} = S_{vs} = S_{vf} = \{S_{loc}, S_{loc}, \ldots, S_{loc}\}$. There are four process expressions modeling as below.

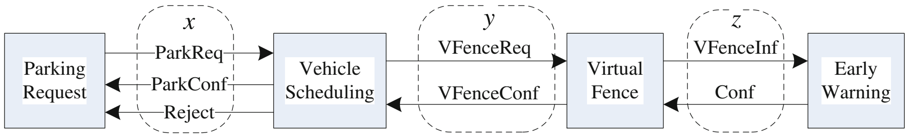

Fig. 6. Business process of the Parking Electric Fence

(1) Parking Request

When a parking request message is sent in t_0, answer must received in $t_0 + 3$ seconds (unit below the same).

$$\begin{aligned} P_{pr} =& Int(t_0, 0)Loc[S_{pr}]\bar{x}\langle ParkReq\rangle.(Int(t_0, 3)Loc[S_{pr}]x(ParkConf) + \\ & Int(t_0, 3)Loc[S_{pr}]x(Reject)) \end{aligned} \tag{2}$$

(2) Vehicle Scheduling

Within $t_1 + 2$, this MCS service must react to parking request in time. Once the congestion level reaches a certain one, virtual fence request should be sent within $t_2 + 1$. Then virtual fence confirmation has to be received in $t_3 + 1$.

$$\begin{aligned} P_{vs} =& Loc[S_{vs}]x(ParkReq).(Int(t_1,2)Loc[S_{vs}]\overline{x}\langle ParkConf\rangle \\ & + Int(t_1,2)Loc[S_{vs}]\overline{x}\langle Reject\rangle). \\ & [msg = ParkConf](Int(t_2,1)Loc[S_{vs}]\overline{y}\langle VFenceReq\rangle. \\ & Int(t_3,1)Loc[S_{vs}]y(VFenceConf)) \end{aligned} \tag{3}$$

(3) Virtual Fence

Within $t_4 + 1$ of receiving virtual fence request, electric fence will be set up. After that, each car in benchmark region would receive early warning messages in $[t_5, t_5 + 30]$ ($t_5 = t_5 + 30$, increasing constantly). Let us take *CarA* inside of benchmark region for example. Confirmation message from Car*A* has to be got in $t_{Ae} + 30$ (t_{Ae} is virtual fence information sending time).

$$\begin{aligned} P_{vf} =& Loc\left[S_{vf}\right]y(VFenceReq).Int(t_4,1)Loc\left[S_{vf}\right]\overline{y}\langle VFenceConf\rangle. \\ & !\ (Int(t_5,30)Loc\left[S_{vf}\right]\overline{z}\langle VFenceInf\rangle.Int(t_{Ae},30)Loc\left[S_{vf}\right]z(Conf)) \end{aligned} \tag{4}$$

(4) Early Warning

Within $t_{Ar} + 10$ (t_{Ar} isvirtualfence information receiving time) is virtual fence information receiving time), early warnings would be sent to Car*A*.

$$P_{ew} = !\ (Loc[S_{ew}]z\langle VFenceInf\rangle.Int(t_{Ar},10)Loc[S_{ew}]\overline{z}(Conf)) \tag{5}$$

We express Parking Electric Fence MCS service composition as *W*. $W = P_{pr}|P_{vs}|P_{vf}|P_{ew}$. On the basis of formal verification of Sect. 3, temporal and spatial attributes of actual MCS components are introduced into *W*. Then from *W* being deadlock or not, we can judge whether this MCS service composition can reach final state. After that, the validity can be verified, and errors could be recognized early.

In practice, park electric fence runs normally in six months' test, which validates the correctness of analysis results. So this case study shows that the formal verification for MCS components composition is reasonable.

5 Conclusion

In this paper, a service-oriented architecture and a formal verification method are proposed for MCS components composition. The case study demonstrates that they are of innovative and practical. The future works will focus on two aspects as follows. (1) Achieve reusing by adding process adapter for MCS services which can't meet space and time constraints. (2) Make optimal decision by taking further study on action-time function and construct time state space.

References

1. Olaleye, S.B., Ishan, R.: A review on enhancing storage space of smartphone using virtualization. In: 2015 International Conference on Futuristic Trends on Computational Analysis and Knowledge Management, pp. 739–743. IEEE Press, Noida (2015)
2. Fernando, N., Loke, S.W., Rahayu, W.: Mobile cloud computing: a survey. J. Future Gener. Comput. Syst. **29**, 84–106 (2013)
3. Dinh, H.T., Lee, C., Niyato, D., Wang, P.: A survey of mobile cloud computing: architecture, applications, and approaches. J. Wirel. Commun. Mob. Comput. **13**, 1587–1611 (2013)
4. Satyanarayanan, M., Bahl, P., Caceres, R., Davies, N.: The case for VM-based cloudlets in mobile computing. J. Pervasive Comput. **8**, 14–23 (2009)
5. Zhang, X., Kunjithapatham, A., Jeong, S., Gibbs, S.: Towards an elastic application model for augmenting the computing capabilities of mobile devices with cloud computing. J. Mob. Netw. Appl. **16**, 270–284 (2011)
6. Erl, T.: Service-Oriented Architecture: Concepts, Technology, and Design. Prentice Hall PTR, New Jersey (2005)
7. Tan, J.Q., Kavulya, S., Gandhi, R., Narasimhan, P.: Visual, log-based causal tracing for performance debugging of mapreduce systems. In: 30th International Conference on Distributed Computing Systems, pp. 795–806. IEEE Press, Genoa (2010)
8. Gutierrez-Garcia, J.O, Sim, K.-M.: Agent-based service composition in cloud computing. In: Kim, T.-H., Yau, S.S., Gervasi, O., Kang, B.-H., Stoica, A., Ślęzak, D. (eds.) GDC/CA 2010. CCIS, vol. 121, pp. 1–10. Springer, Heidelberg (2010)
9. Milner, R.: Communicating and Mobile Systems: The π-Calculus. Cambridge University, Cambridge (1999)
10. Wang, P., Yang, L., Li, G.W., Gao, X.: A novel substitution judgment method for mobile cloud computing application system components. In: 6th International Conference on Cloud Computing Technology and Science, pp. 911–916. IEEE Press, Singapore (2014)
11. Egenhofer, M.J., Herring, J.R.,: A mathematical framework for the definition of topological relationships. In: 4th International Symposium on Spatial Data Handling, Zurich, pp. 803–813 (1990)