Research on Growth Scheduling and Processing in Cyber-I Modeling

Huang, Wei

(出版者 / Publisher)
法政大学大学院情報科学研究科
(雑誌名 / Journal or Publication Title)
法政大学大学院紀要. 情報科学研究科編 / 法政大学大学院紀要. 情報科学研究科編
(巻 / Volume)
11
(開始ページ / Start Page)
1
(終了ページ / End Page)
6
(発行年 / Year)
2016-03-24
(URL)
https://doi.org/10.15002/00012877

Research on Growth Scheduling and Processing in Cyber-I Modeling

Wei Huang
Graduate School of Computer and Information Sciences
Hosei University
Tokyo 184-8584, Japan
wei.huang.tm@stu.hosei.ac.jp

***Abstract*—With the progressive development of information and communication technologies, we are now facing a new world called hyper world that is composed by the cyber world and the physical world with various digital explosions including data, connectivity, service and intelligence. Therefore, Cyber-I has been proposed, which is a real individual's counterpart in cyberspace, is to create a unique, digital, comprehensive description for every human being. In order to provide appropriate services to our individuals, Cyber-I's model has been defined as a dynamic one, and can be built by utilizing an increasing amount of personal data with adaptive methods. A growable Cyber-I model is necessary to achieve the adaptation for successive approximation to its corresponding real individual (Real-I). This paper presents our research and development of an adaptable system, called Cyber-I growth modeling system. The system is trying to (1) arrange the schedule of Cyber-I's growth according to data and time, (2) manage the quantity of raw data that is involved in a specific growth process, (3) generate the Cyber-I model data with appropriate growth forms, and (4) record the information of a Cyber-I model's growth process into a log file in personal database.**

Keywords—Cyber-I; modeling; personal data; adaptation; growth; process; system; schedule; log; model data

I. INTRODUCTION

With the development of the Internet and computer technologies, a digital explosive era is coming silently and rapidly to our daily life. The increasing amount of personal data will be generated in all corners in our life during using Web services, mobile terminals, wearable devices and various sensors. We are hardly to know how our personal data will be used in somewhere, and to discern what the necessary or useless data is for our better life. Therefore, Cyber-I, short for Cyber-Individual, is proposed as a real individual's counterpart in cyber world. It can assist a real individual (Real-I) to solve various problems and handle different services that may be impossible to tackle by ourselves [1]-[2].

Cyber-I has been regarded as a digital clone for our human individual from the personal data and related behaviors in the mind/thinking [3]. A Cyber-I has a life cycle like a living entity from birth, growth to death [4]. In order to provide appropriate services to us, it is extremely significant to constantly capture our behaviors and accurately analyze increasing personal data with adaptive methods. Therefore, a growable Cyber-I model is necessary to proximate its Real-I. Zhang and others have defined three growth forms for a Cyber-I model to become bigger, higher and closer [5].

In this research, our main objective is to study when a Cyber-I model grows and how it grows, namely what are proper growth scheduling and associated processing. To study the growth process, it is necessary to build a Cyber-I growth modeling system (CGMS) to manage the growth scheduling and associated processing with vast personal data. In this system, one of the crucial problems is when the growth process will be activated. A function module, which is called growth scheduler, has been designed based on data-driven and time-driven mechanisms to activate data processing. Moreover, another crucial problem is how to grow by processing the personal data. To solve this problem, a model controller has been designed to manage the model growth. Other modules, such as growth process logger and data processing marker, have also been contained in this system.

This paper is to present our research on the Cyber-I growth scheduling and processing. The remainder of this paper is organized as follows. Section II covers related work to describe relationships between our Cyber-I modeling system and related studies. Section III gives an overview on the Cyber-I growth modeling system. Section IV describes the growth scheduling and data processing based on data arrival time and amount. Section V explains the process of model controlling and logging in modeling growths. Section VI shows the key techniques used in system implementation and a case study of the model growth process. Conclusions about the system and necessary to improve the system in the future are shown in the last section.

II. RELATED WORK

A user model represents a collection of personal data associated with a specific user [6]. Therefore, it is the basis for any adaptive changes to the system's behavior. User modeling is a subdivision of human-computer interaction and describes the process of building up and modifying a user model, whose main goal is customization and adaptation of systems to the user's specific needs [7].

Supervisor: Prof. Jianhua Ma

A generic user modeling server for adaptive Web systems (GUMSAWS) is designed and implemented to reach the goals of generality, extend-ability, and replaceability, which acts as a centralized user modeling server to assist several adaptive Web systems concurrently [8]. It offers the user modeling functions of building up a user profile, and storing, retrieving, updating and deleting entries. It provides the user modeling tasks of inferring user property values and providing adaptive services with recommendations according to users' interests.

Benefit from the device technology, lifelong user modeling has the potential to help people achieve their long-term goals such as healthy weight, increasing physical activity or lifelong learning such as learning a foreign language [9]. It is important for people to monitor and reflect on their progress, adjust their actions and behavior based on real evidence over the long term [10].

Lifelong Machine Learning (LML) is a significant component of machine learning that is a subfield of computer science that evolved from the study of pattern recognition and computational learning theory in artificial intelligence [11]. Therefore, LML considers systems that can learn many tasks from one or more domains over its lifetime, and its goal is to sequentially retain learned knowledge and to selectively transfer that knowledge when learning a new task so as to develop more accurate hypotheses or policies.

Our Cyber-I growth modeling system focuses on neither some restricted domains nor applications. Comparing to general user modeling, the personal data that is collected from various devices is abundant so that the Cyber-I model will cover more aspects of characteristics and emphasize more on processing the collected personal data for individuals.

III. CYBER-I GROWTH MODELING SYSTEM

Fig. 1 shows an overview of Cyber-I growth modeling system (CGMS) as well as basic system functions and related modules in the system.

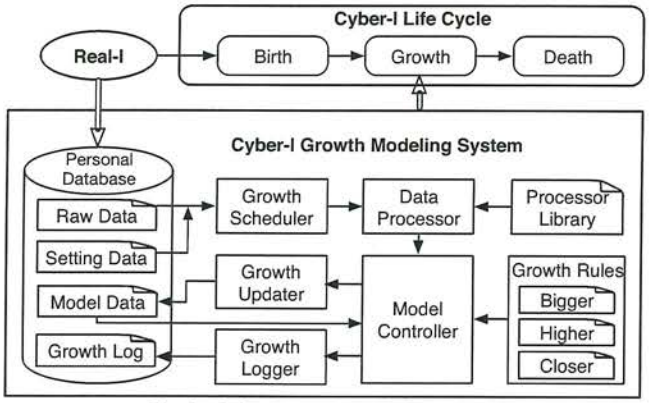

Fig. 1. Cyber-I growth modeling system

A. Cyber-I Life Cycle and Growth Forms

Similar to the human being's life cycle, a Cyber-I's life cycle is defined as to represent each specific stage of Cyber-I's life. A Cyber-I life cycle has three stages, which are birth stage, growth stage and death stage [4]. A Cyber-I growth form are specific ways to control the growth process, which represents that the whole growth process should utilize at least one or more ways in growth modeling. It includes three growth forms that are bigger, higher and closer [5].

The 'bigger' means that the model could grow to cover more aspects of an individual just as a tree grows with more branches. The 'higher' means an abstract refinement of description based on a Real-I's interest, behavior and trait. The 'closer' is a process, during which the model is successively adjusted to reduce errors generated in the previous stages, or adapt to the abrupt changes in the user's attributes.

B. Personal Database

Personal Database is an important component in our CGMS, which stores various kinds of personal data in a Mongo DB. In the database there are four kinds of personal data, which are raw data, setting data, model data and growth log. The raw data is the data for a single person, which is obtained from various devices, sensors and applications according to individual's behaviors, locations or situations. The setting data is defined as a set of system configuration data used in CGMS. The model data is a type of special data in growth modeling system, which is generated by growth processing as an output by utilizing various kinds of raw data. The growth log is a file that records events that occur in a Cyber-I growth process, which represents that the events are successfully executed or suspended.

C. Funcation Modules in Model Growth Process

The system is serving as a common place for Cyber-I's growth modeling with various personal data, which includes five main functional modules that are growth scheduler, data processor, model controller, growth updater and growth logger. Relations of the function modules are shown in Fig. 1.

The growth scheduler is required during arrivals of various data types from different data sources to the database, which is expected to make a decision to decide whether an growth processing execution starts or not. The data processor is a module that receives arrived data as well as explicit processor information from growth scheduler, and executes associated algorithms that are extracted from processor library according to processor information. The model controller is to generate new model data associated with processed data received from the data processor as well as the growth rules including bigger, higher and closer. The growth updater is a module that updates model data in the model database as well as marks data processing state with data process record in the personal database. The growth logger is a module by which a specific log file will be written into growth logs as a record associated with the information about the growth process.

IV. GROWTH SCHEDULING AND DATA PROCESSING

The growth scheduler is to manage the schedules in processing raw data, which is to decide when the growth process should be activated according to the raw data and idle time. There are mainly two scheduling mechanisms designed in the growth scheduler, which are data-driven scheduling

(DDS) and time-driven scheduling (TDS). An example to illustrate the two scheduling mechanisms is shown in Fig. 2.

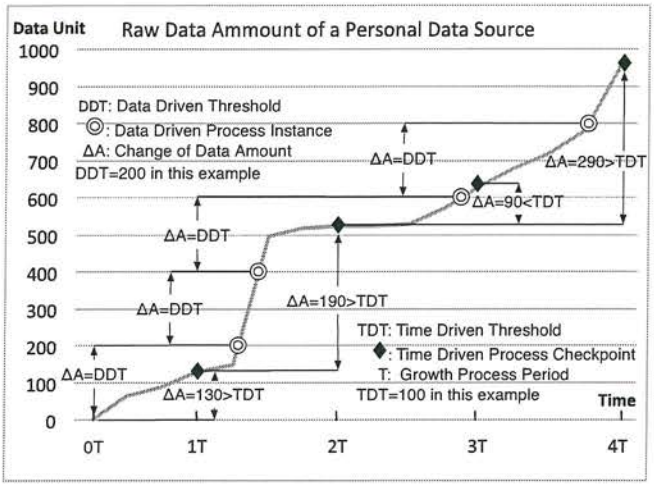

Fig. 2. Data-driven scheduling (DDS) and time-driven scheduling (TDS)

The data-driven scheduling, i.e., DDS, is a schedule mechanism based on the amount of personal data arrived in the raw database, which has two key parameters that are a data-driven threshold (DDT) and a changed data amount (ΔA). The DDT represents the minimum data amount to start growth processing for a specific personal data source in each data-driven process instance. The value of DDT is capable of being dynamically adjusted in order to adapt to the specific growth process state. ΔA is represented for the change of data amount, which means the accumulation quantity of whole unprocessed raw data counted from last data-driven process instance in personal database. That is, the data processing with DDS will be depended upon a temporal instance when the ΔA exceeds DDT for a data source.

The time-driven scheduling, i.e., TDS, is defined as a schedule mechanism based on the time of the growth process that is recorded in the system, which also has two key parameters that are time-driven threshold (TDT) and growth process period (T). TDT is the lowest amount of data to start growing processing for a specific data type. The value of TDT can be dynamically adjusted in order to adapt the specific growth process state. T is represented for the growth process period, which means the time that is calculated from last time of processing to this one. T is another adaptive value that can be dynamically adjusted by growth scheduler. That is, the data processing with TDS will check the data change periodically in a period T and detect these data sources whose data amount changes ΔAs exceed TDT.

Two illustrative examples are shown above in Fig. 2. The upper-left part shows DDS states. As we see, the DDT of this kind of raw data is temporally set to 200 units by the growth scheduler, which represents the top limitation of data processing amount in one data-driven process instance is 200 units. When the value ΔA exceeds (or equals) 200 units, the growth process will be activated immediately. A processor is selected in processor library as a specific algorithm to process the selected data.

The lower-right part in Fig. 2 shows the TDS states. First, the value of TDT is temporally set to 100 units in this example, which means the lowest limitation of the amount of processing raw data for one growth process period. Then, a countdown timer that is a specialized type of clock for measuring time intervals to work on. When the time interval set to one growth process period has expired, data checking process will be activated at this time checkpoint. If ΔA<TDT (100), growth processing will postpone for the period T to the next process checkpoint since the lack of enough amount of unprocessed raw data. If ΔA>=TDT (100), a growth processing will be triggered immediately. A processor is selected in the processor library as a specific algorithm to process the selected data.

A. Data-Driven Scheduling

To implement DDS, there are a data listener and two specific calculation algorithms, which are ΔA calculation algorithm and the DDT calculation algorithm. A flowchart of DDS is presented in Fig. 3.

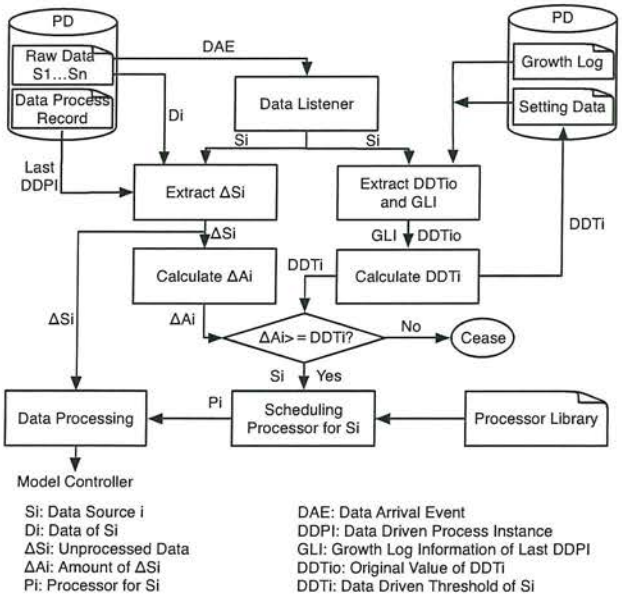

Fig. 3. Flowchart of data-driven scheduling

The most crucial problem of scheduling is to find out when the growth process will go into operation. In order to cope with this problem, first of all, when a batch of raw data that is sent through a personal data source is received by database, the information of this event is detected by a data listener that accurately records the data type and the data quantity by analyzing the event information in database. The analyzed information involved in data type and data quantity is dispatched to the ΔA calculation and the DDT comparison.

The ΔA calculation algorithm is a sophisticated one that tries to figure out the quantity of unprocessed raw data, which includes not only the raw data of the current event but also the remained unprocessed data with the same data type from the time point of last data-driven process instance. The algorithm tries to calculate the value of ΔA according to the feature of a specific data type since it is not generally reasonable to

calculate the data amount with simply counting total bytes of the data.

Another important algorithm in DDS is called DDT calculation algorithm, which is utilized to figure out the value of data-driven threshold that is completely different according to the prescribed data type. That means each data type should have a particular value of DDT. When the function of DDT calculation receives a message that includes the information of data type from data listener, the DDT calculation algorithm is supposed to extract the original value of DDT from setting data as well as the growth log information in the previous data-driven process instance from the growth log database. With the two kinds of parameters, a particular value of DDT for the prescribed data type is accurately calculated by analyzing the growth log files and combining the current situations of data with the data utilization record.

As the result of these two calculation algorithms above, the values of ΔA and DDT are received by a function of comparison, which tries to compare the value of ΔA with DDT and make a decision about the next step of the process. If ΔA < DDT, the growth process is supposed to be ceased since the lack of enough unprocessed data until the next event of data arrival. If ΔA >= DDT, the growth process is assumed to continue to execute the next step, which will send a message to the function of processor scheduling and dispatch the unprocessed data to the data processor module.

B. Time-Driven Scheduling

In the TDS, there are a timer and two calculation algorithms similar to the ones in DDS. A flowchart of time-driven scheduling is shown in Fig. 4.

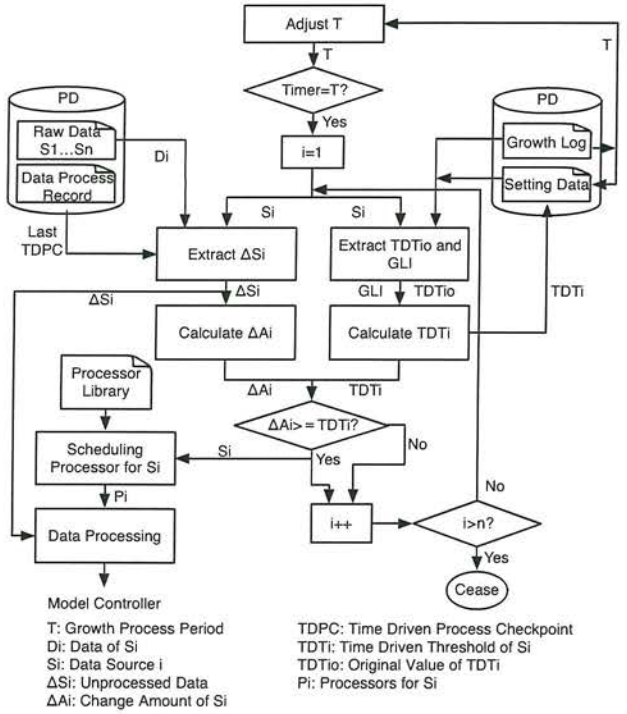

Fig. 4. Flowchart of time-driven scheduling

In order to figure out when the growth data processing will be executed, an effective method is to set up a timer that is supposed to count down from a moment, which the number of counting down equals to a time period we called T. That means when the timer counts down the time of T, the TDS algorithm will be activated immediately. In addition, the value of T will be dynamically adjusted according to the situation of the growth process.

Then, TDS will begin to check the state of raw data in personal database. For each kind of raw data, TDS tries to execute the ΔA calculation algorithm and TDT calculation algorithm. The ΔA algorithm has been described in DDS. The TDT calculation algorithm is quite different with DDT, which is to figure out the lowest amount limitation of raw data for each specific data type. The algorithm will extract the original value of TDT that is the initial TDT value of a specific data type as well as the previous growth log information of TDT for this data type. Then, the TDT calculation algorithm is supposed to figure out a new value of TDT by analyzing the data utilization record and the initial TDT value combining with the current situation of unprocessed raw data.

The value of ΔA and TDT will be received by a comparison function as a sophisticated checking process, which is expected to make a decision of whether the growth process will be carried out or not according to the comparison result. If ΔA < TDT, this data type will not be chosen to take part in the next growth process. Therefore, the TDS process comes back to the finished point of timer and tries to examine another data type that has unprocessed raw data. If ΔA >= TDT, the growth process will continue to execute the next step, which will send a message to the function of processor scheduling and dispatch the unprocessed data to data processor module. The loop of data examination is also executed and the process will come back to the finished point of timer. The key for TDS is to set suitable TDTs for different kinds of raw data.

V. GROWTH PROCESS IN CYBER-I MODELING

The growth process is a procedure to utilize the processed data to generate or update model data by controller, growth updater and growth logger as shown in Fig. 5.

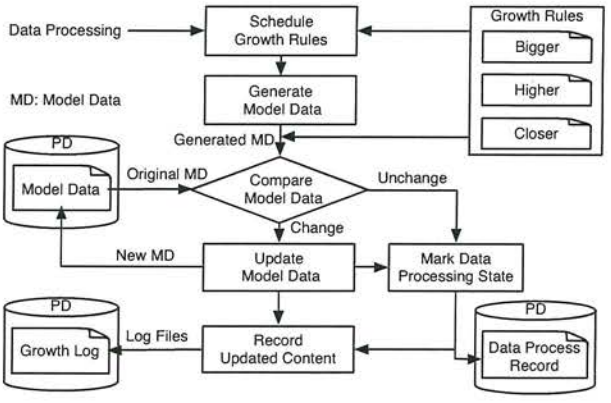

Fig. 5. Cyber-I model control, update and log

The model controller is the core module of the growth process, which is to analyze the processed data associated with growth rules for becoming bigger, higher and closer, and the generate the model data. The growth updater is a data interaction module, which is to update the existing model with the newly generated one. The growth logger is the module that tries to record all events in the growth process into a log file that is stored in personal database.

According to the information of the processed data, the control module is expected to schedule the growth forms and the frequency of each growth form, which is meant to choose one or more appropriate algorithms for generating model data and allocate the time of algorithm execution. Then, the model controller tries to generate the model data. The generated data is preparing to be estimated by a comparison function. The comparison function is to compare the generated model data with original model data that has the same classification with the new one. If the generated model data has the same characteristics with corresponding original model data or has no more characteristics, the Cyber-I model data will not be changed by these newly generated model data. If the generated model data has more characteristics comparing with the original model data, the corresponding original model data will be updated in the personal database. After that, if the decision is to update the model data in the database, the new model data is supposed to be dispatched to growth updater as the result of model controlling. Moreover, a data usage record of these raw data used for this growth process will be marked in the personal database, which is a new starting instance to calculate the next data change amount for the next growth processing.

According to the classification of updating data, the growth updater tries to locate the position in database and update Cyber-I model data to the new one. After the process of database, a message will be dispatched to growth logger.

The growth logger will be activated by the message, which includes comprehensive parameters and the state of data usage. The record is supposed to be reorganized in a normalization form, which corresponds to the form of log file. So, the log file will be made and inserted into personal database in case of data corruption and loss when some unexpected accidents happen.

VI. System Implementation and Case Study

The Cyber-I growth modeling system is based on the browser/server architecture where Tomcat acts as the server. The system is implemented by the Java EE platform associated with JSP and Apache struts2 framework, which is an open-source Web application framework for developing Java EE Web applications. Mongo DB is chosen as the database for the storage of data, which usually gains high performance in dealing with the massive personal data and heterogeneous data types/media. HTML5, JavaScript and jQuery cooperate with each other to enrich the system functions in a dynamic way. The architecture of system implementation is depicted in Fig. 6.

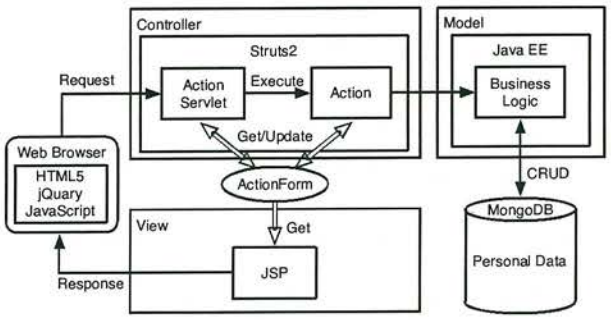

Fig. 6. Architecture of system implementation

As a case study, a specific personal data is shown as an example of the growth process. The personal data that is gathered from Twitter is used as the specific example. One item of the tweet data from Twitter, which has arrived at the Mongo DB, is given in Fig. 7.

```
{
                _id: 55462f4b317e1f175033826f
          Data Type: Tweet String
         Categories: Behavior
     Sub-categories: Hobby
            Content: Google visit
          Frequency: 8 times a day
      Creation Time: Mon May 11 13:23:53 JST 2015
}
```

Fig. 7. An example of twitter data

When a batch of raw data has been received, the data arrival event will be detected by the data listener of growth scheduler. Then, the information of raw data that has the same data type with the new arrival data will be extracted. The calculation function of ΔA and DDT will execute the algorithm of tweet string. Depending on the feature of tweet, the calculation algorithm of ΔA will count the number of items and the quantity of bytes for each item. Then, the average value of each item will be calculated. Meanwhile, a new DDT value will also be calculated by analyzing the situation of previous processing.

A comparison function will be triggered immediately after the values of ΔA and DDT have been calculated. We suppose that the $\Delta A = 240$ and $DDT = 200$. So the $\Delta A > DDT$, and then a message will be sent to the data processor module associated with the processing data, which contains the data type of the processing data. According to tweet string, the function will search the processor library and try to find out a corresponding processor for tweet string data. Next, the tweet string processor will process the algorithm for these tweet data. The processed data will be dispatched to the model controller that is supposed to analyze the feature of the processed data. In this case study, it is necessary to analyze the specific content including data type, categories, sub-categories, content and frequency. Depending on the characteristic of the data, the growth rule scheduling function will call the corresponding growth form that is bigger in this case as the algorithm for the processed data. Therefore, the new model data will be generated by bigger algorithm. A possible result of generated model data is given in Fig. 8.

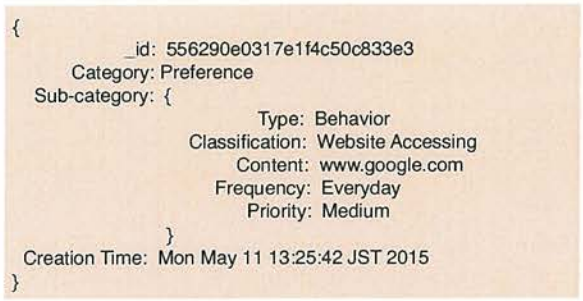

Fig. 8. A possible result of generated model data

Once the generated model data is received, a comparison function will be executed to compare the original model data with the new one. As a result, we can know that the original model data of the corresponding classification has no such the content like Google yet. So the update module will modify the model data to a new one. A new Cyber-I model data is organized as Fig. 9.

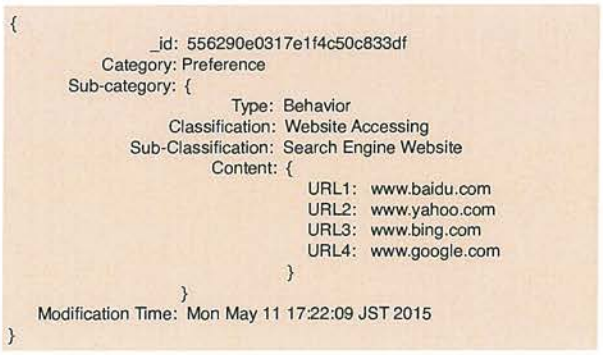

Fig. 9. An new Cyber-I model data

As a result, the new Cyber-I model data will be dispatched to the growth updater. In the growth updater module, according to the values of category, classification, and sub-classification, the module will locate the position in the database. Then, the new model data will be updated and saved into the personal database. What's more, a message will be recorded by the growth updater, which includes the whole information of current growth process.

The growth logger will receive the message and transform it into the normalization form specified by the log file. Then, the growth related events would be recorded in the corresponding position of the log file in personal database. Meanwhile, the information of growth process will be also received by a marking function that sets the data processed mark in the corresponding position of the raw data sequence. By now, an entire growth process from detection of new data arrival, checking the changed data amount, processing the data, updating the model, recording the growth event, and marking the data has been shown.

VII. Conclusion and Future Work

This research has been focused on the mechanism and procedure of Cyber-I growth modeling with various personal data collected from different data sources. The Cyber-I growth modeling system developed is capable of dealing with the scheduling and processing of growth, which is (1) to arrange the schedule of Cyber-I's growth according to data and time; (2) to manage the quantity of raw data involved in a specific growth process; (3) to select a corresponding processor to handle raw data; (4) to generate Cyber-I model data with one or more growth forms; (5) to record the growth process with a log file in personal database.

The design of the Cyber-I growth modeling system is just a basic architecture of processing and managing the Cyber-I's growth. Therefore, there is much work remaining to improve the system in the following aspects. (1) A coordination mechanism should be found in order to arrange the priority rate of data driven scheduling and time driven scheduling. (2) The calculation algorithms of ΔA, DDT and TDT with more details should be designed in future to make the system more complete. (3) The adaptive threshold and period are supposed to be implemented in order to adjust the corresponding parameters promptly. (4) According to the concept of growth forms, the related algorithms should be developed concretely. (5) The procedure of the model growth process should be further refined and optimized accurately. (6) More APIs and application tools should be developed in order to let more developers and researchers access to the Cyber-I growth modeling system.

References

[1] W. Jie, M. Kai, W. Furong, H. Benxiong, and M. Jianhua, "Cyber-I: Vision of the individuals counterpart on cyberspace," IEEE International Conference on Dependable Autonomic and Secure Computing (DASC 09), pp. 295-302, 2009.

[2] J. Ma, J. Wen, R. Huang and B. Huang, "Cyber-Individual meets brain informatics," IEEE Intelligent Systems, vol. 26, No.5, pp. 30-37, September/October 2011.

[3] J. Ma and R. Huang, "Digital explosions and digital colones", in Proc. of the IEEE International Conference on Internet of People (IoP-2015), Beijing, 2015.

[4] J. Ren, J. Ma, R. Huang, et al, "A management system for cyber individuals and heterogeneous data", in Proc. of the 11th IEEE International Conference on Ubiquitous Intelligence and Computing (UIC2014), 2014.

[5] S. Zhang, J. Ma, R. Huang, et al, "Growable Cyber-I's modeling with increasing personal data," in Proc. of the International Conference on Advances in Computing, Control and Networking (ACCN 2015), 2015.

[6] A. Kobsa, "Supporting user interfaces for all through user modeling," In Proceedings of the Sixth International Conference on Human-Computer Interaction, vol. 1, pp. 155-157, 1995.

[7] V. Marco, B. Nadia, and E.Z. Elod, "A survey on user modeling in multi-application environments," The 3rd Int'l Conf. on Personalized Mechanisms, Technologies and Services, pp. 111- 116, 2010.

[8] J. Zhang and A. A. Ghorbani, "Gumsaws: A generic user modeling server for adaptive Web systems," in Fifth IEEE Annual Conference on Communication Networks and Services Research (CNSR'07), pp. 117–124, 2007.

[9] J. Key, and B. Kummerfield, "Lifelong user modeling goals, issues and challenges," Proceedings of the Lifelong User Modelling Workshop at UMAP, 2009, pp. 27-34.

[10] L. Tang and J. Kay, "Lifelong user modeling and meta-cognitive scaffolding: Support Self Monitoring of Long Term Goals," UMAP Workshops, pp. 2-4, 2007.

[11] S.L. Daniel, Q. Yang, and L. Li. "Lifelong machine learning systems: Beyond learning algorithms," AAAI Spring Symposium on Lifelong Machine Learning, 2013.