A Snapshot Browsing Model for Web-based Surveillance System in Heterogeneous Computing Environment

Sheng-Tun Li*
Department of Information Management
National Kaohsiung First University of Science and Technology
1 University Road,
Yenchao, Kaohsiung
Taiwan 824, R.O.C.
Tel: 886-7-6011000-4111
Fax: 886-7-6011042
stli@ccms.nkfust.edu.tw

Shu-Ching Chen
School of Computer Science
Florida International University
Miami, FL 33199, USA
chens@cs.fiu.edu

Mei-Ling Shyu
Department of Electrical and Computer Engineering
University of Miami
Coral Gables, FL 33124, USA
shyu@miami.edu

Abstract

In this paper, we present the *multimedia augmented transition network (MATN)* model that is the underlying semantic model for the snapshot browsing in a distributed surveillance services (DSS) system. The MATN model provides the visualization of control structure and a good programming data structure for implementation of the proposed snapshot browsing system. MATNs are left to right models that are used to model browsing sequences. In order to simplify the network structure, one subnetwork is constructed for each time duration. The MATN and its subnetworks depict the structural hierarchy of the snapshot browsing graph. By using the subnetwork structure, searching time will be heavily reduced by traversing and searching the corresponding subnetworks selected by the user without traversing the whole network. The DSS system is a distributed object-based system that integrates the next generation Internet-based applications in the architecture level and in the system design level. The DSS system establishes a firm basis for supporting a variety of potential cost-effective applications in a plug-and-play manner. By incorporating the MATN model and utilizing the MATN control structures, complicated browsing operations involving user interactions can be easily implemented for the DSS. A detailed example that illustrates how to use the MATN model to model the snapshot browsing in surveillance monitor of the DSS system is also included.

*To whom all correspondences should be addressed.

Keywords: Distributed multimedia system, Augmented transition networks, Distributed surveillance services (DSS) system.

1 Introduction

Recently, multimedia information systems have emerged as a fruitful area for research due to the recent progress in high-speed communication networks, large capacity storage devices, digitized media, and data compression technologies over the last few years. Multimedia information has been used in several applications including manufacturing, medicine, education, business, entertainment, media-on-demand, video conference, visual query methodology, etc. As more information sources become available in multimedia systems, the development of abstract semantic models for video, audio, text, and image data becomes very important. An abstract semantic model has two requirements. First, it should be able to effectively provide visualization of control structure for multimedia systems. Hence, the semantic model can model the hierarchy of visual contents so that users can browse and decide on various scenarios they want to see. Second, it should be a good programming data structure for implementation to control multimedia systems.

Many semantic models for multimedia systems have been proposed in the literatures. However, these existing semantic models either have difficulty to model user interactions, or are too complicated for users to understand the control structures [2, 6, 15, 16, 20]. For this purpose, we propose a semantic model called the augmented transition network (MATN) [7, 8] to model snapshot browsing in a distributed surveillance services (DSS) system. A security/surveillance service is an important multimedia application that allows users to remotely watch real-time snapshots on the spot in a universal manner, to retrieve historical suspicious scenarios, to access the heterogeneous media, and even to receive urgent signals in the form of email or pager. Particularly, it is a great challenge to provide a cross-platform solution for capturing video or still images since it involves the driving of low-level devices [5]. Because of the seamless integration of the World Wide Web (WWW) and the emerging Java technology, the universal accessibility to the surveillance service becomes possible.

In our previous study [13, 14], a Java-centric distributed object-based DSS system was developed to discuss the interoperability issues in heterogeneous computing environments. The "bridging technologies" [4], in particular, the seamless integration of CORBA/RMI/Java/Web [18, 12] for supporting integrated surveillance services, were adopted. In our design, the Java programming language was chosen since it can serve as the infrastructure for building network centric enterprise applications [3]. Though Java was designed for small customer devices initially, it becomes more prevalent when it is used to develop distributed applications on the WWW. For example, an information technology organization in Sun Microsystems called SUNIR built a java-centric multi-tier service-based environment for supporting more than twenty thousand employees in 180 locations in 55 countries [10, 19].

This DSS system consists of three components – the Capturer Server, Surveillance Server, and Client Monitor. The Capturer Server is equipped with an H.263-compliant video camera. The snapshot signals on the spot are periodically grabbed by Snapshot Grabber into the form of an integer array. The Surveillance Server plays the middleware between Capturer Server and Surveillance Monitor to alleviate the overhead of snapshot grabbing in Capturer Server. The Surveillance Monitor offers a variety of services such as browsing snapshots on the spot and image archives, storing images monitored to the local repository on the client side, obtaining a local hardcopy of the image upon request, sending an alert pager call under suspicious situations, and forwarding images with messages via SMTP Client. Applications such as computer room management and laboratory monitoring on campus, intensive care in hospitals, traffic monitoring, and small-scale distance learning systems can make use of the proposed DSS system. Currently, a prototype of the DSS system is now being run to support computer room management at NKFUST (National Kaohsiung First University of Science and Technology).

In this paper, the MATN model for snapshot browsing of the Surveillance Monitor in the DSS system is discussed in details. The MATN model meets the two requirements of an abstract semantic model for multimedia information systems since it provides a simple visualization control structure and a good programming data structure for implementation of the proposed snapshot browsing system. An MATN can be represented diagrammatically by a labeled directed graph and it consists of a finite set of state nodes connected by the labeled directed arcs. The arcs in an MATN represent the time flow from one state node to another. An arc represents an allowable transition from the state node at its tail to the state node at its head, and the labeled arc represents the transition function. MATNs are left to right models that are used to model browsing sequences. In order to simplify the network structure, one subnetwork is constructed for each time duration for the snapshot browsing. Also, the MATN model has the capability to model user interactions. User interactions are represented by using more than one outgoing arc with different arc labels in the decision state node. Each user selection will lead to a corresponding subnetwork.

The inputs for MATNs and subnetworks are modeled by multimedia input strings. Multimedia input strings provide an efficient means for iconic indexing of the time durations and time instants in multimedia browsing. By using the subnetwork structure, searching time will be heavily reduced since only the necessary subnetwork will be traversed and searched based on the user selections without going through the whole network. Therefore, the MATN and its subnetworks can depict the structural hierarchy of the snapshot browsing graph for the DSS system. By utilizing the MATN control structures, complicated browsing operations involving user interactions can be easily implemented for the DSS.

This paper is organized as follows. Section 2 briefly describes the DSS system along with its three key components. The discussion of how to use the MATN model to model and control snapshot browsing in surveillance monitor is presented in Section 3. Section 4 gives a detailed example to explain the operations of using the MATN model and its subnetworks to model the snapshot browsing for the Surveillance Monitor in the DSS system. Conclusion is presented in Section 5.

2 The DSS System

In our previous study, an object-based DSS system is built for supporting surveillance services [13, 14]. The system architecture is shown in Figure 1. As can be seen from this figure, there are three key components – the Capturer Server, Surveillance Server and Client Monitor. The following three subsections briefly describe these three components.

For the detailed description of the DSS system, please see [14].

2.1 Capturer Server

Capturer Server is equipped with an H.263-compliant video camera. The CPU-intensive requirement of video signal acquisition necessitates that Capturer Server runs on a dedicated machine to capture encoded image signals. Users are allowed to manually set the grabbing rate, the JPEG quality and the timestamp format on the snapshot. Snapshot Grabber in Capturer Server is currently implemented in MFC (Microsoft Foundation Class). To be interoperable with the Surveillance Server component, Snapshot Grabber is wrapped by CORBA IDL. IDL, a declarative language and independent of operating systems as well as ORBs, provides a limited range of data types, for example, char, boolean, short, long, struct, sequence, etc. but visual message types like video and images are excluded. Therefore, the snapshot signals on the spot are periodically grabbed by Snapshot Grabber into the form of an integer array.

2.2 Surveillance Server

Surveillance Server plays the middleware between Capturer Server and Surveillance Monitor to alleviate the overhead of snapshot grabbing in Capturer Server. It is purely implemented in Java so it can be installed on any platforms. Surveillance Server provides diverse services ranging from Image Composition, Web Service, Pager Service, Visitor Log service to Archiving Service. Image Composer passes snapshot signals delivered from Capture Server in the format of an integer array to Snapshot Feeder for providing the function of viewing images on the client. In addition, Image Composer constructs an encoded JPEG image from the integer array and sends it to Archiving Server for being stored in the image repository, Image Archives. The system manager of Surveillance Server may decide how long to keep for these snapshots and establish the schedule to purge archives.

2.3 Surveillance Monitor

Surveillance Monitor offers a variety of services; namely, browsing snapshots on the spot and image archives, storing images monitored to the local repository on the client side, obtaining a local hardcopy of the image upon request, sending an alert pager call under suspicious situations, and forwarding images with messages via SMTP Client. The components of Surveillance Monitor are also fully coded as Java applets and can run on any Java-compliant Web browsers, e.g. Netscape Communicator 4.07. To relieve the restriction of the sandbox security model, the applet is digitally signed and is packaged as a JAR format for reducing the loading time and easing the installation of mobile code on the client. When downloading the signed applet, the end-user is prompted a message for confirming the certificate with the applet. The end-user may choose granting, denying or viewing the certificate. Figure 2 gives the outlook of the user interface of Surveillance Monitor. There are several time instants provided on the right sub-window of Figure 2 to allow the users to select the particular snapshots they want to browse. The time instants selected by the user are highlighted as shown in Figure 2.

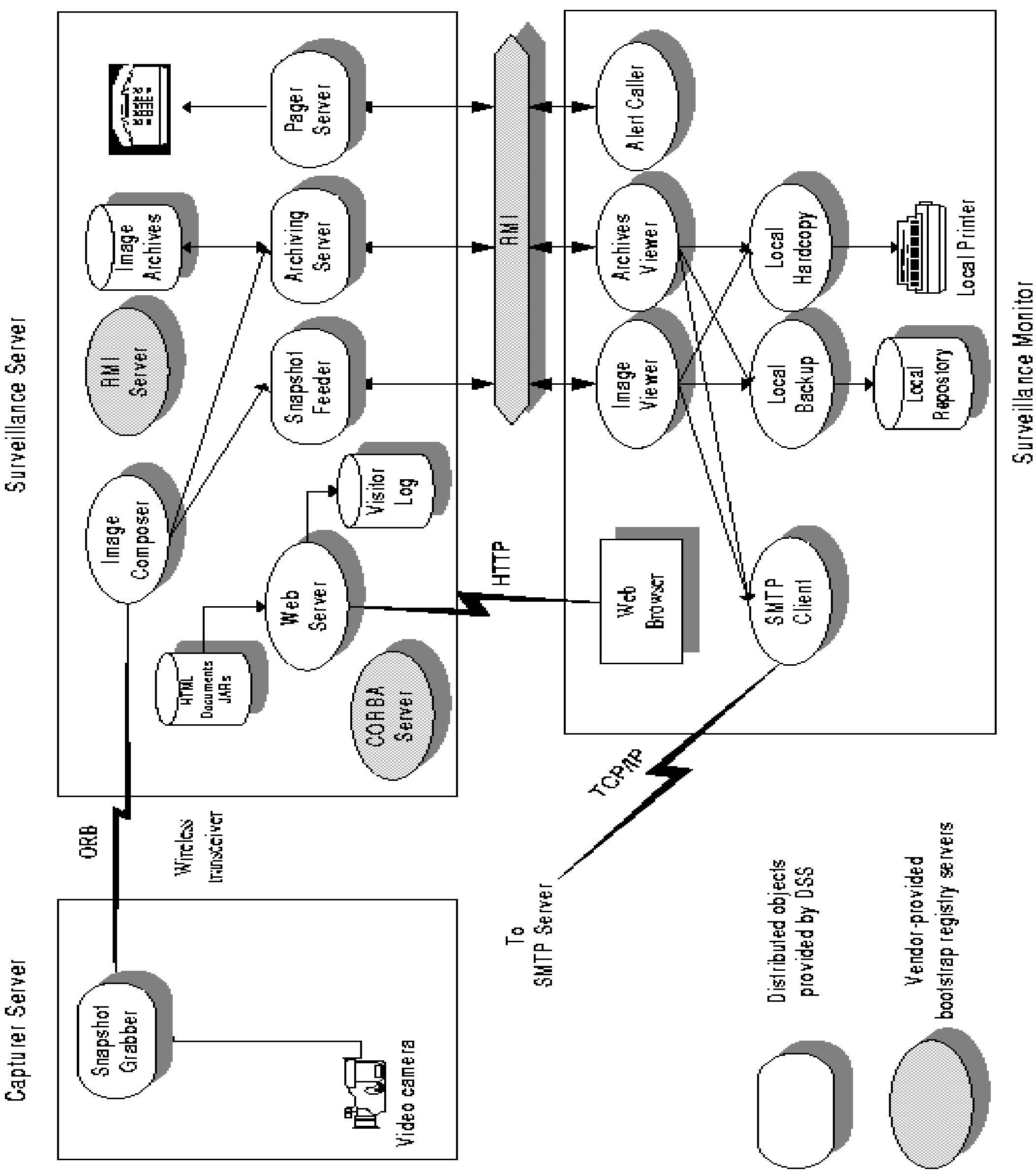

Figure 1: DSS architecture.

Figure 2: The outlook of user interface control in Surveillance Monitor.

3 Using MATN to Model and Control Snapshot Browsing in Surveillance Monitor

3.1 Functionalities of Surveillance Monitor

The design principals and functionality of the Surveillance Monitor are discussed as follows.

- **Delegation-based Visual User Interfaces:**

  In the design of the Surveillance Monitor, a delegation-based event model that has been proposed since JDK 1.1 for larger-scale GUI development is adopted. The reason for adopting the delegation-based event model is that the earlier event-loop and inheritance-based AWT model requires to write a loop, wait for some events to occur, and then take the associated actions. In the new model, one simply needs to register event handlers (called event listeners) with an AWT (Abstract Window Toolkit) component. Whenever some events come up on the component, its associated event listener will be invoked. In addition, a set of friendly icons has been implemented for the AWT components to empower end-users to interact with the system in a drag-and-drop manner,

- **Image Viewer:**

  Image Viewer is a sub-component within the Surveillance Monitor (as shown in Figure 1). It periodically receives snapshot signals in the integer array format provided by Snapshot Feeder through RMI, constructs a JPEG image, and then displays the image on the Surveillance Monitor frame. There are two routines that can be called by Image Viewer. The Local Backup Routine is used to store images on the local (or networking) repository; whereas the Local Hardcopy routine is used to obtain a hardcopy of an image.

- **Archives Viewer:**

  Archives Viewer is a sub-component within the Surveillance Monitor (as shown in Figure 1). Time-stamped image archives carried by Archiving Server over RMI slide-by-slide can be examined by the Archives Viewer. Though network bandwidth or computer processing power makes the delivery of live videos expensive, some interesting images can still be marked up by using a mouse so a simulated video could be obtained for browsing (as shown in Figure 2). Similar to the Image Viewer, Archives Views can save and print suspicious images locally.

- **Alert Caller:**

  Alert Caller is a sub-component within the Surveillance Monitor (as shown in Figure 1). Alert Caller provides users a convenient way to send an urgent signal to the manager on-duty via pager or e-mail when an emergency occurs. The Pager Server powered in Java Communications API in Surveillance Server routes the pager number sent by Alert Caller through RMI to the RS232 serial port on the server machine. The signal recipient then may remotely sign onto DSS and inspect the spot being monitored.

- **Java MIME-Compliant SMTP Service:**

  SMTP Client in Surveillance Monitor provides a Java MIME-compliant reusable package (as shown in Figure 1). SMTP (Simple Mail Transfer Protocol) defined in RFC 821 is a standard e-mail sending protocol for a TCP/IP-based network. A client can simply send a message to an SMTP server by creating a socket connection to port 25 on the server. RFC 821 requires the SMTP data be 7-bit ASCII characters whereas RFC 822 describes the format and some of the message content semantics but contains no content structure specification. RFC 1521 remedies the limitation of ASCII-based messages by specifying an invertible encoding/decoding scheme (e.g., BASE64, QUOTED-PRINTABLE, 7BIT and 8BIT) for message data types. The Multipurpose Internet Mail Extension (MIME) standard, defined in RFCs 2045 - 2049, complements RFCs 821/822 defining of the message content structure and support of the encoding of different types of media data (image, audio, and video) for transmission over SMTP protocol. Currently, the SMTP Client can transmit two MIME text message types – text/plain and test/html, and two image types – image/jpeg and image/gif encoded in the Base64 Content-Transfer-Encoding format as it is preferred over quoted-printable for binary data [17].

3.2 The MATN Model

The augmented transition network (ATN), developed by Woods [21], has been used in natural language understanding systems and question answering systems for both text and speech. In our previous studies, we used multimedia augmented transition network (MATN) as a semantic model to model multimedia presentations [7, 8], multimedia database searching, the temporal, spatial, or spatio-temporal relations of various media streams and semantic objects [8], and multimedia browsing [9]. The MATN model is based on the ATN with the major modifications to model multimedia information.

- **The Structure of MATNs:** At the heart of MATN is a finite state machine (FSM). The FSM is a simple transition network that consists of nodes and directed arcs connecting them. The nodes correspond to states and the arcs represent the transitions from state to state. Each arc is labeled with a symbol whose input can cause a transition from the state at the tail of the arc to the state at its head. This feature makes FSM have the ability to model a presentation from the initial state to some final states or to let users watch the presentation with pre-defined sequence. Therefore, the FSM has difficulty to model user interactions.

  MATN differs from an FSM in that it permits recursion so that MATN is a *recursive transition network*. This property can eliminate the weakness of the FSM as mentioned above. A recursive transition network is similar to an FSM with the modifications as follows. In MATNs, states are represented by circles with the state name inside. The state name is used to indicate the presentation or subnetwork being displayed (to the left of the slash) and which time instant snapshot image has just been displayed. The state name in each state can tell us all the events that have been accomplished so far. Based on the state name, we can know how much of the presentation has been displayed. When the control passes to a state, it means all the events before this state are finished. A state node is a breaking point for two different events.

  All states are given names which are then allowed as part of labels on arcs in addition to the normal input symbols. Based on these nonterminal symbols, subnetworks are created. That is, each nonterminal symbol consists of a subnetwork which can be used to model detailed information for the corresponding nonterminal symbol. In our design, a subnetwork is constructed for each time duration of the snapshots in the DSS system. User interactions are represented by using more than one outgoing arc with different arc labels in the decision state node. Each user selection will lead to a corresponding subnetwork to reduce the unnecessary overhead caused by traversing the whole network.

- **Arc Types in MATNs:** The arc types together with the notation need to be defined. We adopted the following notation and definition as in [1].

  - **Push** arc: succeeds only if the named network can be successfully traversed. The state name at the head of arc will be pushed into a stack and the control will be passed to the named network.
  - **Pop** arc: succeeds and signals the successful end of the network. The topmost state name will be removed from the stack and become the return point. Therefore, the process can continue from this state node.
  - **Jump** arc: always succeeds. This arc is useful to pass the control to any state node.

  These three types of arcs are used to control the snapshot browsing sequences. Browsing provides users the opportunity to view information rapidly since they can choose the content relevant to their needs. It is similar to the table of contents and the index of a book so that readers have a general idea by simply looking at them. The advantage is that users can quickly locate the interesting topic and avoid the sequential and time-consuming process.

- **Multimedia Input Strings as Input for MATNs:** Multimedia input strings adopt the notations from regular expressions [11]. Regular expressions are useful descriptors of patterns such as tokens used in a programming language. Regular expressions provide convenient ways of specifying a certain set of strings. In this study, multimedia input strings are used to represent the browsing sequences of the snapshot browsing in the DSS system. A multimedia input string goes from left to right, which can represent the time sequence of a multimedia browsing.

  Two notations $\mathcal{L}$ and $\mathcal{D}$ are used to define multimedia input strings:

  $\mathcal{L}$ = {d, t} is the set whose member d represents the time duration and member t denotes time instant.
  $\mathcal{D}$ = {0, 1, ..., 9} is the set consisting of the set of the ten decimal digits.

  Definition 1: Each input symbol of a multimedia input string contains one or more time durations (time instants) which are displayed at the same time interval. A time duration is a string which begins with a letter d subscripted by a string of digits in $\mathcal{D}$. A time instant is a string which begins with a letter t subscripted by a string of digits in $\mathcal{D}$. For example, d_1 (or t_1) represents a time duration (or time instant) and its identification number is one.

  - **Concurrent:** The symbol "&" between two time durations (or time instants) indicates these two time durations (or time instants) are displayed concurrently for users to make selections. For example, (d_1&d_2) represents d_1 and d_2 being displayed concurrently to allow users to make selections.
  - **Looping:** $m^+ = \bigcup_{i=1}^{\infty} m^i$ is the multimedia input string of positive closure of m to denote m occurring one or more times. We use the "+" symbol to model loops in a multimedia browsing to let some part of the presentation be displayed more than once.
  - **Contiguous:** Input symbols which are concatenated together are used to represent a multimedia presentation sequence and to form a multimedia input string. Input symbols are displayed from left to right across time sequentially. ab is the multimedia input string of a concatenated with b such that b will be displayed after a is displayed.
  - **Alternative:** A multimedia input string can model user selections by separating input symbols with the "|" symbol. So, $(a|b)$ is the multimedia input string of *a or b*.
  - **Ending:** The symbol "$" denotes the end of the presentation.

An example by using the MATN and the multimedia input string to model the snapshot browsing will be discussed in Section 3.

4 An Example to Use MATN to Model Snapshot Browsing in Surveillance Monitor

Table 1 lists part of the time instants shown in Figure 2. Four time durations, denoted by d_1 to d_4, are used to capture those time instants. Here, each time instant

Table 1: Part of the time durations, time instants, and the corresponding selections for the time instants in Figure 2.

Duration	**Time instant Symbol**	**time instant**	Selected or not?
	t_1	1998/09/15 10:44:50	No
d_1	t_2	1998/09/15 10:44:55	No
	t_3	1998/09/15 10:45:00	No
	t_4	1998/09/15 10:45:05	Yes
d_2	t_5	1998/09/15 10:45:10	Yes
	t_6	1998/09/15 10:45:15	Yes
	t_7	1998/09/15 10:45:20	Yes
d_3	t_8	1998/09/15 10:45:25	Yes
	t_9	1998/09/15 10:45:30	Yes
	t_{10}	1998/09/15 10:45:35	Yes
d_4	t_{11}	1998/09/15 10:45:40	Yes
	t_{12}	1998/09/15 10:45:45	Yes

is represented by a time instant symbol (e.g., t_4), and each time duration consists of three time instants. The fourth column of Table 1 indicates whether each time instant is selected by the users. A time instant is selected if the users want to browse the corresponding snapshot. For example, the snapshots for the time instants with time instant symbols t_5 to t_{12} are selected for browsing. This specified browsing sequence will be kept by the MATN model to guide the traversing steps. Figure 3 shows the MATN and its four subnetworks to model the Surveillance Monitor browsing scenario of Table 1. Figure 3(a) is the MATN and Figure 3(b) to Figure 3(e) are the four subnetworks for the four durations, respectively.

In Figure 3(a), the state name $P/$ indicates the presentation P is displayed. The outgoing arc symbol is $d_1\&d_2\&d_3\&d_4$ which means there are four time durations for users to select. The "&" symbols are used to represent d_1, d_2, d_3, and d_4 are displayed concurrently to users for selection. d_1, d_2, d_3, and d_4 are nonterminal symbols and are the beginning state names of four subnetworks as shown in Figure 3(b)-(e). The state name P/d_1 is to indicate the scenario that users select time duration d_1, the subnetwok with beginning state name $d_1/$ has been traversed, and snapshop images in this subnetwork are already displayed. In Figure 3(b), the state name d_1/t_1 represents subnetwork d_1 finishes the displaying of snapshot image at time instant t_1. Arc symbol "Jump" will pass the control directly to the pointing state.

Part of the trace of MATN of the example browsing sequence (shown in Table 1) is presented in Table 2. Though in Table 1, duration d_2, d_3, and d_4 are selected, it is assumed that duration d_2 is selected for browsing and stops for simplicity reason. This detailed trace in Table 2 is used to illustrate how the MATN and its subnetworks model the Surveillance Monitor browsing scenario.

Step 1: The current state is in $P/$ where P represents the starting of the browsing. The input symbol is $d_1\&d_2\&d_3\&d_4$. This input symbol denotes the time instants in durations d_1 to d_4 are displayed concurrently for users to make selection (as shown on the right sub-window of Figure 2). This input symbol is read so that users can select a time instant belonging to a duration.

Step 2: Duration d_2 is selected so the input symbol d_2 is read. Since d_2 is a subnetwork name, the state name P/d_2 is pushed into a stack. A stack follows the last-in-first-out (LIFO) policy which only allows retrieving the topmost state name first.

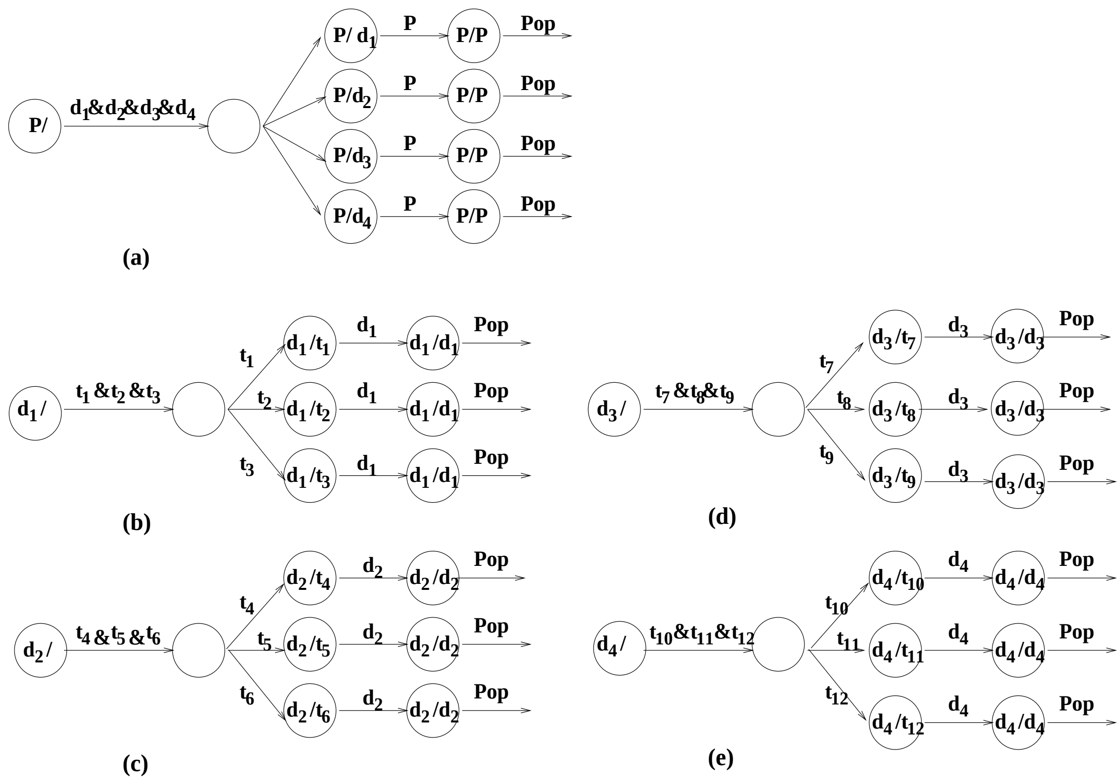

Figure 3: (a) MATN to model the Surveillance Monitor browsing scenario of Figure 2 with respect to four durations. (b) to (e) are the subnetworks for durations d_1 to d_4, respectively.

Step 3: The control is passed to the subnetwork with starting state name $d_2/$ (as shown in Figure 3(c)). The input symbol is $t_4\&t_5\&t_6$. This input symbol denotes the time instants in duration d_2 are displayed concurrently for users to make selection.

Step 4: The current state is in $d_2/t_4\&t_5\&t_6$. The input symbol t_4 is read which indicates that the snapshot image for the time instant t_4 is selected for browsing.

Step 5: The current state is in d_2/t_4. The input symbol d_2 is read. Since d_2 is a subnetwork name, the state name d_2/d_2^1 is pushed into the top of the stack. The superscript 1 on d_2/d_2^1 is used to indicate that this is the first time the state name d_2/d_2 is pushed into the stack.

Step 6: The control is passed to the beginning of subnetwork with starting state name $d_2/$ and the input symbol $t_4\&t_5\&t_6$ is read again.

Step 7: The current state is in $d_2/t_4\&t_5\&t_6$. The input symbol t_5 is read which indicates that the snapshot image for the time instant t_5 is selected for browsing.

Step 8: The current state is in d_2/t_5. The input symbol d_2 is read. Since d_2 is a subnetwork name, the state name d_2/d_2^2 is pushed into the top of the stack. The superscript 2 on d_2/d_2^2 is used to indicate that this is the second time the state name d_2/d_2 is pushed into the stack.

Table 2: The trace of MATN for the browsing sequence of duration d_2 in Figure 3.

Step	Current State	Input Symbol	Backup States
1	$P/$	$d_1\&d_2\&d_3\&d_4$	
2	$P/d_1\&d_2\&d_3\&d_4$	d_2 (Push)	P/d_2
3	$d_2/$	t_4	P/d_2
4	d_2/t_4	Jump	P/d_2
5	$d_2/$	t_5	P/d_2
6	d_2/t_5	Jump	P/d_2
7	$d_2/$	t_6	P/d_2
8	d_2/t_6	Jump	d_2/d_2^2 d_2/d_2^1 P/d_2
9	$d_2/$	$t_4\&t_5\&t_6$	d_2/d_2^2 d_2/d_2^1 P/d_2
10	$d_2/t_4\&t_5\&t_6$	t_6	d_2/d_2^2 d_2/d_2^1 P/d_2
11	d_2/t_6	d_2 (Push)	d_2/d_2^3 d_2/d_2^2 d_2/d_2^1 P/d_2
12	$d_2/$	Pop	d_2/d_2^3 d_2/d_2^2 d_2/d_2^1 P/d_2
13	d_2/d_2^3	Pop	d_2/d_2^2 d_2/d_2^1 P/d_2
14	d_2/d_2^2	Pop	d_2/d_2^1 P/d_2
15	d_2/d_2^1	Pop	P/d_2
16	P/d_2	P (Push)	P/P
17	$P/$	Pop	P/P
18	P/P	Pop	
19	Stop		

Step 9: The control is passed to the beginning of subnetwork with starting state name $d_2/$ and the input symbol $t_4\&t_5\&t_6$ is read again.

Step 10: The current state is in $d_2/t_4\&t_5\&t_6$. The input symbol t_6 is read which indicates that the snapshot image for the time instant t_6 is selected for browsing.

Step 11: The current state is in d_2/t_6. The input symbol d_2 is read. Since d_2 is a subnetwork name, the state name d_2/d_2^3 is pushed into the top of the stack. The superscript 3 on d_2/d_2^3 is used to indicate that this is the third time the state name d_2/d_2 is pushed into the stack.

Step 12: The *Pop* arc signals the successful end of the subnetwork. The topmost state name d_2/d_2^3 is removed from the stack and becomes the return point.

Step 13: The current state is in d_2/d_2^3. The outgoing *Pop* arc removes the topmost state name d_2/d_2^2 from the stack which becomes the return point.

Step 14: The current state is in d_2/d_2^2. The outgoing *Pop* arc removes the topmost state name d_2/d_2^1 from the stack which becomes the return point.

Step 15: The current state is in d_2/d_2^1. The outgoing *Pop* arc removes the topmost state name P/d_2 from the stack which becomes the return point.

Step 16: The current state is in P/d_2. The input symbol P is read. Since P is a network name, the state name P/P is pushed into the top of the stack.

Step 17: The current state is in $P/$. The outgoing *Pop* arc removes the topmost state name P/P from the stack which becomes the return point.

Step 18: The current state is in P/P. The outgoing *Pop* arc removes the topmost state from the stack. Since the stack is empty now, the control stops.

Step 19: The browsing sequence stops.

Steps 1 to 19 show the scenario that only duration d_2 is selected. Similiar steps can be applied for the durations other than d_2.

5 Conclusion

In the last few years, much attention has been focused on multimedia information technologies and applications. For example, with the integration of the World Wide Web (WWW) and the Java technology, a security/surveillance service system becomes an important application since the universal accessibility to such a system becomes possible. In our previous study, we have developed a Java-centric distributed object-based DSS system whose goals are to provide a cost-effective, reliable, flexible, open and universal environment for multimedia message exchanges yet accommodate the overwhelming proliferation of online information and the heterogeneous systems. Under such a system, the development of abstract semantic is required in order to have a control mechanism for the security/surveillance services.

There are two requirements for an abstract semantic model. First, it should be able to effectively provide visualization of control structure for multimedia systems. Second, it should be a good programming data structure for implementation to control multimedia systems. In this paper, we presented a semantic model called the MATN model that is the underlying semantic model for the snapshot browsing in the DSS system. A detailed example that illustrates how to use the MATN model and its multimedia input string to model the snapshot browsing in surveillance monitor of the DSS system is also included in this paper. As we can see from this example, the MATN model meets both the requirements for serving as an abstract semantic model for the snapshot browsing in the DSS system. Unlike existing semantic models for multimedia systems which either have difficulty to model user interactions, or are too complicated for users to understand the control structures, the MATN model provides a visualization structure for the snapshot browsing in the DSS system. The MATN model is also a systematic implementation structure to develop snapshot browsing in the DSS system. In our design, one subnetwork is constructed for each time duration for the snapshot browsing, which simplifies the network structure. Hence, the MATN and its subnetworks have the capabilities to depict the structural hierarchy of the snapshot browsing graph for the DSS system.

Overall speaking, the MATN model provides the visualization of control structure and a good programming data structure for implementation of the proposed snapshot browsing system. By utilizing the MATN control structures, complicated browsing operations involving user interactions can be easily implemented for the DSS. In addition, the use of the subnetwork structure will heavily reduce the searching time since only the necessary subnetwork will be traversed and searched based on the user selections without the need to traverse the whole network.

Acknowledgments

This project is supported in part by the Innovation and Incubation Center of National Kaohsiung First University of Science and Technology under contract 87B22. The DSS project receives a Java Programming Award, Ministry of Economic Affairs, TAIWAN, R.O.C., 1998.

References

[1] James Allen, *Natural Language Understanding*, The Benjamin/Cummings Publishing Company, Inc., 1995.

[2] Y.Y. Al-Salqan and C.K. Chang, "Temporal Relations and Synchronization Agents," *IEEE Multimedia*, pp. 30-39, Summer 1996.

[3] S. Ashok and V. K. Bansal, "Java: Network-centric Enterprise Computing," *Computer Communications*, Vol. 20, pp. 1467-1480, 1998.

[4] S. Baker, V. Cahill, and P. Nixon, "Bridging Boundaries: CORBA in Perspective," *IEEE Internet Computing*, ,pp. 52-57, Sept-Oct. 1997.

[5] P. Cadolini, F. Davoli, G. Marescotti, and P. Maryni, "Developing a Distance Learning System using Java Applets," *Procs. of IEEE Global Telecommunications Conference*, pp. 63-66, 1996.

[6] H.J. Chang, T.Y. Hou, S.K. Chang, "The Management and Application of Teleaction Objects," *ACM Multimedia Systems Journal*, Vol. 3, pp. 228-237, November 1995.

[7] S.-C. Chen and R. L. Kashyap, "Temporal and Spatial Semantic Models for Multimedia Presentations," *1997 International Symposium on Multimedia Information Processing*, pp. 441-446, December 1997.

[8] S.-C. Chen and R. L. Kashyap, "A Spatio-Temporal Semantic Model for Multimedia Presentations and Multimedia Database Systems, *IEEE Transactions on Knowledge and Data Engineering*, accepted for publication, 2000.

[9] S.-C. Chen, S. Sista, M.-L. Shyu, and R. L. Kashyap, "Augmented Transition Networks as Video Browsing Models for Multimedia Databases and Multimedia Information Systems," *11th IEEE International Conference on Tools with Artificial Intelligence (ICTAI'99)*, pp. 175-182, November 9-11, 1999, Chicago, IL, U.S.A.

[10] A. Gupta, C. Ferris, Y. Wilson, and K. Venkatasubramanian, "Implementing Java Computing: Sun on Architecture and Applications Deployment," *IEEE Internet Computing*, March-April, pp. 60-64, 1998.

[11] S.C. Kleene, "Representation of Events in Nerve Nets and Finite Automata, Automata Studies," *Princeton University Press*, Princeton, N.J., pp. 3-41, 1956.

[12] JavaSoft Company, "Java Remote Method Invocation," http://www.javasoft.com/products/jdk/rmi/.

[13] S.-T. Li, W.-S. Chen, H.-C. Chang, and C.-M. Lin, "Building Integrated Surveillance Services Using Java/CORBA Bridging Technologies in Heterogeneous Computing Environments," *the 11th International Conference on Systems Research, Informatics and Cybernetice*, Aug, 1999, Baden-Baden, Germany.

[14] S.-T. Li and W.-S. Chen, "A Java-centric Distributed Object-based Paradigm for Surveillance Services and Visual Message Exchange," accepted for publication, *Int'l Journal of Visual Languages and Computing*, 1999.

[15] C.C. Lin, J.X., S.K. Chang, "Transformation and Exchange of Multimedia Objects in Distributed Multimedia Systems," *ACM Multimedia Systems Journal*, Vol. 4, pp. 12-29, February 1996.

[16] T.D.C. Little and A. Ghafoor, "Synchronization and Storage Models for Multimedia Objects," *IEEE J. Selected Areas in Commun.*, Vol. 9, pp. 413-427, Apr. 1990.

[17] Network Working Group, MIME (Multipurpose Internet Mail Extensions) Part One: Mechanisms for Specifying and Describing the Format of Internet Message Bodies, http://www.isi.edu/in-notes/rfc1521.txt.

[18] OMG, "CORBA," http://www.omg.org/corba/.

[19] SUN Press Release, "SUN and IBM Collaborate on Java Media Technologies," Internet World, http://www.javasoft.com/pr/1998/10/pr981007.html, October 7, 1998.

[20] T. Wahl and K. Rothermel, "Representing Time in Multimedia Systems," *Proc. Int'l Conf. on Multimedia Computing and Systems*, CS Press, Los Alamitos, Calif., pp. 538-543, 1994.

[21] W. Woods, "Transition Network Grammars for Natural Language Analysis," *Comm. of the ACM*, **13**, October 1970, pp. 591-602.