

## Object-oriented Structuring of Real-Time Autonomous Decentralized Systems

K. H. (Kane) Kim

Department of Electrical & Computer Engineering  
University of California  
Irvine, CA 92697, U.S.A.  
Kane@Ece.Uci.Edu

(Position paper)

The application of the autonomous decentralized system (ADS) structuring to real-time (RT) computing applications, especially in large-scale complex applications, is expected to show steady increase in coming years. This is because the ADS structuring leads to a high degree of maintainability.

A promising cost-effective approach to realizing RT ADS's is to use the emerging technology for RT distributed component/object structuring. One such technology being explored by the author and his research collaborators is the the *Time-Triggered Message-Triggered Object* (TMO) structuring scheme [Kim94, Kim97b]. The TMO scheme is intended to facilitate RT distributed software engineering in a form which software engineers in the vast business software field can adapt to with small efforts. It is a syntactically simple and natural but semantically powerful extension of the conventional object structuring approaches. The scheme facilitates uniform structuring of (1) both RT and non-RT distributed application systems, (2) both control computer systems and their application environment simulators, and (3) requirement specifications and system designs arising at various phases of system engineering.

At the same time the TMO scheme is aimed for allowing system engineers to confidently produce certifiable RT ADS's for safety-critical applications, i.e., the systems possessing the reliability which will be demanded by the general public in the 21st century. The scheme enables a great reduction of the designer's efforts in guaranteeing timely service capabilities of distributed computing application systems. Its support tools can be based on well-established OO programming languages such as C++ and JAVA and on ubiquitous commercial RT operating system kernels or even on NT. In addition, the TMO scheme facilitates an attractively simple approach to parallel and distributed RT simulation [Kim94, Kim97b]. It was devised to bring about a major improvement in the design productivity and product reliability in complex distributed system engineering.

TMO extends the conventional objects with several unique features including:

(TF1) Spontaneous method (SpM):

The TMO contains a new type of methods, *time-triggered (TT-) methods*, also called the *spontaneous methods*

(SpM's), in addition to the conventional *service methods* (SvM's). The SpM executions are triggered when the RT clock reaches specific values determined at design time whereas the SvM executions are triggered by service request messages from clients. Moreover, actions to be taken at real times *that can be determined at the design time* can appear only in SpM's.

(TF2) *Basic concurrency constraint (BCC)*:

Under this rule, SvM's cannot disturb the executions of SpM's and the designer's efforts in guaranteeing timely service capabilities of TMO's are greatly simplified. Basically, *activation of an SvM triggered by a message from an external client is allowed only when potentially conflicting SpM executions are not in place*. An SvM is allowed to execute only when no SpM that accesses the same portion of the Object Data Store (ODS) (i.e., a group of data members) to be accessed by this SvM has an execution time window that will overlap with the execution time window of this SvM. However, the BCC does not stand in the way of either concurrent SpM executions or concurrent SvM executions.

(TF3) A *time window* imposed on each output action and completion of a method:

By advertising these time window specifications to the designers of potential client objects, the designer of the server TMO guarantees the timely services of the object. Before determining the time window specifications, the server object designer must make sure that with the available *object execution engine* (hardware plus operating system) the server object can be implemented such that the output actions are performed within the time windows.

Several preliminary experimental validations undertaken by the proposers and other research organizations [Kim94, Kim97b, Sho98] have shown the great potential of the TMO structuring in design and implementation of large-scale complex distributed applications in factory automation, traffic control, and military planning/engagement areas. These demonstrations included successful implementations of augmented C++ runtime environments [Kim97a, Sho98].

The TMO structuring scheme eases incorporation of autonomy in the distributed computing systems [Kim95]. First, if an object can freely reject service requests from

clients when the object is in an "uncomfortable" situation, e.g., due to an overload condition created, internal faults, etc., then philosophically one can say that the object is autonomous. In a sense, such capability for structuring *service timing autonomy*, i.e., the autonomy in choosing its times for handling requests from clients, is built into the TMO. . This autonomy is valuable in enabling the easy analysis of the timing behavior of the DCS. In the extreme case of a TMO where SvM's are not allowed to call the SvM's of other objects and thus most calls to SvM's of other objects are made by SpM's, the analysis of the timing behavior of the DCS involves multiple rounds of examining two objects in a client-server relationship at a time. The SvM's in such an object can play the roles of receptionists which receive service requests and then convert them into work orders to be put in a queue which is examined regularly by a master SpM. The master SpM can reject some work orders when the workload is excessive and inform the clients of such rejection.

A TMO may also contain *programmable data-field-channels* [kim95, Kim97a, Mor93]. It is a promising feature of the TMO scheme for facilitating highly abstract (relieving the programmer of the burden of dealing with underlying OS services and network protocols) and yet highly efficient cooperation among remote TMO's. The essence of the data field scheme [Mor93] is to facilitate dynamic creation of *logical multicast channels* and dynamic connection of processes to the logical channels in such a way that the idiosyncracies of the physical communication networks are transparent to the process designer. If the physical communication facility has the broadcast capability, then a logical multicast channel is facilitated by making all processing nodes using the channel broadcast (through the physical communication facility) messages with the headers containing the ID of the channel called the *content code*. The processing nodes connected to the logical channel can see all the messages coming through the physical broadcast facility but will pay attention only to those messages containing relevant content codes. This means that when processes are designed to communicate via the logical multicast channels only, processes can be dynamically relocated without impacting other cooperating processes. The programmable DFC scheme [Kim95] differs from the original data field scheme in that the former allows dynamic flexible connection of processes to the logical channels and supports not only conventional *event messages* but also *state messages* which are based on the distributed replicated memory semantics.

A state message carries information to be stored in a fixed memory location in each receiver corresponding to the ID of the state message [Kim95, Kop97]. Therefore, the ID of a state message represents a group of replicated memory units, each capable of holding the information carried in the state message and belonging to a different receiver. The producer of a state message timestamps the message at the message production time. Each receiver will read the content of its state message memory at its

convenient time. This means that the producer may update the contents of the state message memory units at a higher frequency than that at which a certain receiver reads the content of its state message memory. A state message is thus typically used to share the periodically observed state information about a dynamic object, e.g., temperature of an oven.

In order to further loosen the connection between data producer objects and consumer objects, state message channels can be utilized for providing data directly into the object data stores (ODS's) of consumer objects. Special read-only components connected to state message channels can be included in the ODS. This facilitates the *data acceptance autonomy* of an object, i.e., autonomy in choosing its times for accepting real-time input data. Again, if either member of a producer-consumer pair fails, then the partner is not necessarily blocked. The special read-only components can be viewed as read-only local copies of global shared data space segments (or bulletin board postings).

This author believes that extensive exploitation of both service timing autonomy and data acceptance autonomy is an important subject for further research in the field of RT ADS's.

Moreover, the application domain of agents, which are programs traveling through a network while being aware of its locations, is expected to grow in coming years. In some future time, some RT applications for agents are expected to emerge. For structuring such RT agents, the TMO structuring scheme extended with an appropriate mobile code scheme should serve as a reasonable starting point.

The *software engineering environments* (SEE's) for real-time object-oriented distributed software have not yet been well established. Numerous desirable tools for TMO-structured software engineering are conceivable. The tools most urgently needed but unavailable at this time are those for assisting the TMO designer in the process of determining the response time to be guaranteed.

**Acknowledgment:** The research work reported here was supported in part by the US Defense Advanced Research Project Agency under Contract N66001-97-C-8516 monitored by SPAWAR.

## References

- [Kim94] Kim, K.H. et al., "Distinguishing Features and Potential Roles of the RTO.k Object Model", *Proc. 1994 IEEE CS Workshop on Object-oriented Real-time Dependable Systems (WORDS)*, Oct. 1994, Dana Point, pp.36-45.
- [Kim95a] Kim, K.H., Mori, K., and Nakanishi, H., "Realization of Autonomous Decentralized Computing with the RTO.k Object Structuring Scheme and the H-DF Inter-Process-Group Communication Scheme", *Proc. 1995 IEEE CS's 2nd Int'l Symp. on Autonomous*

- Decentralized (ISADS 95)*, Phoenix, AZ, April. 1995, pp.305-312.
- [Kim95b] Kim, K.H. et al., "A Timeliness-Guaranteed Kernel Model - DREAM Kernel and Implementation Techniques", *Proc. 1995 Int'l Workshop on Real-Time Computing Systems and Applications (RTCSA 95)*, Tokyo, Japan, Oct. 1995, pp.80-87.
- [Kim97a] Kim, K.H., Subbaraman, C., and Bacellar, L., "Support for RTO.k Object Structured Programming in C++", *Control Engineering Practice*, an IFAC journal, Vol.5, No.7, 1997, pp.983-991.
- [Kim97b] Kim, K. H., "Object Structures for Real-Time Systems and Simulators", *IEEE Computer*, Vol. 30, No. 8, August 1997, pp. 62-70.
- [Kop97] Kopetz, H., '*Real-Time Systems*', Kluwer Academic Pub., 1997.
- [Mor93] Mori, K., "Autonomous Decentralized Systems: Concept, Data Field Architecture, and Future Trends", *Proc. IEEE CS Int'l Symp. on Autonomous Decentralized Systems (ISADS 93)*, Mar. 1993, Kawasaki, Japan, pp. 28-34.
- [Sho98] Shokri, E., Crane, P., and Kim, K.H., "An Implementation Model for Time-Triggered Message-Triggered Object Support Mechanisms in CORBA-Compliant COTS Platforms", *Proc. IEEE 1st Int'l Symp. on Object-oriented Real-time dependable Computing (ISORC)*, Kyoto, Japan, April 1998, pp. 12-21.

# PROCEEDINGS

## The Fourth International Symposium on Autonomous Decentralized Systems – Integration of Heterogeneous Systems –

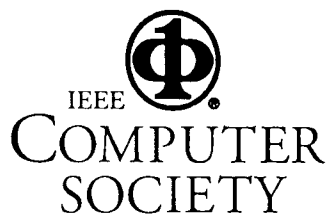
March 21 – 23, 1999  
Tokyo, Japan

*Sponsored by*

IEEE Computer Society  
Information Processing Society of Japan  
The Society of Instrument and Control Engineers of Japan  
The Institute of Electronics, Information and Communication Engineers, Japan

*In Cooperation with*

International Federation for Information Processing  
International Federation of Automatic Control  
OMG  
TINA-C  
Manufacturing Science and Technology Center, Japan



Los Alamitos, California

Washington • Brussels • Tokyo

---