



- [54] **FLEXIBLE STATE SHARING AND CONSISTENCY MECHANISM FOR INTERACTIVE APPLICATIONS**
- [75] Inventors: **Bodhi Mukherjee**, Wappingers Falls; **Srinivas Prasad Doddapaneni**, White Plains, both of N.Y.; **Sumeer Kumar Bhola**, Atlanta, Ga.
- [73] Assignee: **International Business Machines Corporation**, Armonk, N.Y.
- [21] Appl. No.: **09/083,669**
- [22] Filed: **May 22, 1998**
- [51] Int. Cl.⁷ **G06F 15/10; G06F 9/00**
- [52] U.S. Cl. **709/203; 709/303; 709/205; 707/103**
- [58] Field of Search **709/203, 205, 709/303; 707/103**

C. Schuckmann et al., "Designing object-oriented synchronous groupware with COAST", Computer Supported Cooperative Work '96, pp. 30-38, ACM(1996).

(List continued on next page.)

Primary Examiner—Krisna Lim
Attorney, Agent, or Firm—Kevin M. Jordan

[57] **ABSTRACT**

A system, method and computer program storage device providing event and/or state sharing support e.g., for building object-oriented interactive groupware in wide-area distributed environments (such as the Internet). For collaborative applications programmed using events, mechanisms are provided for sharing application-specific events. For example: an event based programming model allows applications to post an event and triggers corresponding ERUs (Event Reaction Unit) in reaction to a received event; preconditions for control activation of ERUs; and event consistency policy objects implement application specified event consistency model. Some policy examples are: a policy in which event order is not guaranteed, but all events are guaranteed to be sent to the ERUs eventually; and a policy that first triggers local ERUs and then posts the event to the server. An out-of-order event is detected using the event notification from the server; and an automatic detection of out-of-order events seen by ERUs in the local workstation in this optimistic event execution model. For applications requiring support for state sharing, an asynchronous model for updating replicated state, which supports atomicity of updates across multiple shared objects is described. Coupled with a flexible marshaling framework, this allows existing application data-structure classes to be easily extended and made shareable. To solve the problem of replica consistency, a novel combination of three mechanisms is used: global locks; detection of incorrect update ordering; and cloning a subset of the shared objects for state re-initialization. To reduce network load due to fine-grained user interaction, a framework for application specified event batching, called Late Event Modification (LEM), enhances the event interface to allow applications to modify the event objects after posting them to the set.

[56] **References Cited**

U.S. PATENT DOCUMENTS

- 5,008,853 4/1991 Bly et al. .
- 5,446,842 8/1995 Schaeffer et al. .
- 5,583,993 12/1996 Foster et al. .
- 5,625,809 4/1997 Dysart et al. .
- 5,634,129 5/1997 Dickinson .
- 5,694,544 12/1997 Tanigawa et al. .
- 5,915,098 6/1999 Palmer et al. 709/249

OTHER PUBLICATIONS

- J. Bacon et al., "Using Events to Build Distributed Applications", University of Cambridge, United Kingdom, 15 pages, (1996).
- H. Chiu et al., "Conferencing Metaphor", IBM Technical Disclosure Bulletin, vol. 36, No. 02, Feb. 1993.
- R. Kordale et al., "Object Caching in a CORBA Compliant System", The USENIX Association, Computing Systems, Title Page, pp. 377-404, vol. 9, No. 4, Fall 1996.
- A. Prakash et al., "Dist View: Support for Building Efficient Collaborative Applications using Replicated Objects", pp. 153-164, CSCW 94- 10/94 Chapel Hill, NC. USA, ACM (1994).

34 Claims, 16 Drawing Sheets

