CLAIMS

1. A method for causing a computing system to process events from a sequence of events that defines a correct order for said events independent from an order in which those events are received over an input device or port, said method including:

defining a first variable,

defining, for said first variable, a first chronology of operations on said first variable associated with received events,

receiving a first event that pertains to said first variable,

executing a first operation on said first variable, wherein said first operation results in a first update of said first chronology,

after having received said first event, receiving a delayed event that pertains to said first variable,

executing a second operation on said first variable, wherein said second operation results in a second update of said first chronology, and

determining whether said first update is valid or invalid,

wherein said delayed event precedes said first event in said sequence,

wherein said first update is based on said first event, and

wherein said second update is based on said delayed event.

2. The method of claim **1**, wherein determining whether said first update is valid or invalid includes determining that said first update is valid.

3. The method of claim **2**, wherein determining that said first update is valid includes determining that said first update is the insertion of a write operation before said first update.

4. The method of claim **2**, wherein determining that said first update is valid includes determining that said delayed event resulted in insertion of a read operation before said first update.

5. The method of claim **2**, wherein determining that said first update is valid includes determining that said delayed event required insertion of an update operation before said first update.

6. The method of claim **2**, wherein determining that said first update is valid includes determining that said first update is not affected by said second update.

7. The method of claim **1**, wherein determining whether said first update is valid or invalid includes determining that said first update is affected by said second update, and rescinding said first update in response to determining that said first update is invalid.

8. The method of claim **1**, wherein determining whether said first update is valid or invalid includes determining that said first update is affected by said second update, causing said first update to be invalid.

9. The method of claim **8**, wherein determining that said first update is affected by said second update includes determining that said delayed

event required insertion of a chronology entry indicating an operation that includes writing said first variable before a chronology entry corresponding to said first update.

10. The method of claim **8**, wherein determining that said first update is affected by said second update includes determining that said delayed event required updating an existing chronology entry indicating an operation on said first variable before a chronology entry corresponding to said first update.

11. The method of claim **1**, wherein said correct order for said sequence of events is partially ordered, and said method further includes causing said sequence of events to be completely ordered.

12. The method of claim **1**, wherein receiving said first event includes receiving said event from a set of completely ordered events.

13. The method of claim **1**, further including

detecting a loss of data in said first chronology,

retrieving a first checkpoint value indicative of a valid value of said first variable in said first chronology,

based on said checkpoint value, retrieving a subset of events from a list of events, said list of events including said first event,

reprocessing said subset of events, thereby reconstructing said data lost in said loss of data, and

revising said first chronology to include said data lost in said

loss of data.

14. The method of claim **1**,

wherein said first chronology includes a plurality of entries corresponding to operations resulting from a plurality of events that are after an entry corresponding to a delayed event operation corresponding to said delayed event,

wherein executing a second operation includes identifying a barrier operation that blocks propagation of invalidity in said plurality of entries, and

revising only those operations from said plurality of operations whose entries are between entries corresponding to said delayed event operation and said barrier operation.

15. The method of claim **14**, wherein identifying a barrier operation includes identifying a write operation as a barrier operation.

16. The method of claim **1**, wherein determining whether said first update is valid includes

identifying an access to said first chronology that must have occurred to correctly execute said first update,

inspecting said first chronology, and

based on said inspection, determining whether said access has occurred.

17. The method of claim **16**, further including

maintaining first information and second information,

wherein said first information identifies an access to said first chronology, and

wherein said second information identifies a range of accesses required to process said first event, and

determining that said first information is within said range of accesses.

18. The method of claim **1**, wherein the computing system includes a plurality of nodes in communication over a network, and said method further includes causing said computing system to execute the steps of

storing said first variable and said first chronology at a first node;

storing a second variable and a second chronology of operations on said second variable at a second node; and

processing at least some events of said sequence of events by performing operations on both said first and second variables, and updating both of said first and second chronologies in response to processing said events.

19. Software, stored in non-transitory form on a computer-readable medium, for causing a computing system to process events from a sequence of events that defines a correct order for said events independent from an order in which those events are received over an input device or port, said software including instructions for causing a computing system to:

define a first variable,

define, for said first variable, a first chronology of operations on said first variable associated with received events,

receive a first event that pertains to said first variable,

execute a first operation on said first variable, wherein said

first operation results in a first update of said first chronology,

receive, after having received said first event, a delayed event that pertains to said first variable,

execute a second operation on said first variable, wherein said second operation results in a second update of said first chronology, and

determine whether said first update is valid or invalid,

wherein said delayed event precedes said first event in said sequence,

wherein said first update is based on said first event, and

wherein said second update is based on said delayed event.

20. A computing system for processing events from a sequence of events that defines a correct order for said events independent from an order in which those events are received over an input device or port, the computing system including:

an input device or port configured to receive said events; and

at least one processor configured to participate in processing of said sequence of events, said processing including

defining a first variable,

defining, for said first variable, a first chronology of operations on said first variable associated with received events,

receiving a first event that pertains to said first variable,

executing a first operation on said first variable, wherein said

first operation results in a first update of said first chronology,

after having received said first event, receiving a delayed event that pertains to said first variable,

executing a second operation on said first variable, wherein said second operation results in a second update of said first chronology, and

determining whether said first update is valid or invalid,

wherein said delayed event precedes said first event in said sequence,

wherein said first update is based on said first event, and

wherein said second update is based on said delayed event.