



Group Communication Specifications: A Comprehensive Study

G. Chockler, I. Keidar, and R. Vitenberg

Présenté par Denis Conan

Juin 2024





Outline

1. Context: Fault-tolerance of view-oriented Group Communication Systems
2. Background and Problem Definition
3. Safety Properties of the Group Membership Service
4. Safety Properties of the Multicast Service
5. Conclusion

1 Context: Fault-tolerance of view-oriented Group Communication Systems

- GCSs provide membership and reliable multicast services
 - Membership service: maintains a list of currently active and connected processes in a group
 - Reliable multicast: delivers messages to the current view members
- GCS applications include state machine (active) replication, distributed transactions, resource allocation, load balancing, collaborative computing
- 1st difficulty of GCS specification = based on agreements, while many agreement problems are known to be intractable in fault-prone async. sys.
 - The specification shouldn't be solvable by trivial (useless) algorithms
 - The specification shouldn't be too strong to implement
- 2nd difficulty of GSC specification = often unclear whether a given property is necessary or sufficient for a certain application



2 Background and Problem Definition

- 2.1 Distributed System Model
- 2.2 External Signature of the GCS service

2.1 Distributed System Model

- Asynchronous distributed system
- Processes may crash and recover; no byzantine failures
- Messages may be lost
- Failures may partition the system into disjoint components
 - Focus on the safety properties
 - Not presented: liveness properties based on unreliable failure detectors that make the components disjoint

2.2 External Signature of the GCS service

Types:

- \mathcal{P} : the set of processes
- \mathcal{M} : the set of application messages
- \mathcal{VID} : the set of view identifiers, partially ordered by the $<$ operator

Interaction with the application¹

- $\text{send}(p, m)$, $\text{recv}(p, m)$, and $\text{view_chng}(p, \langle id, members \rangle \dots)$

Interaction with the environment

- $\text{crash}(p)$ and $\text{recover}(p)$

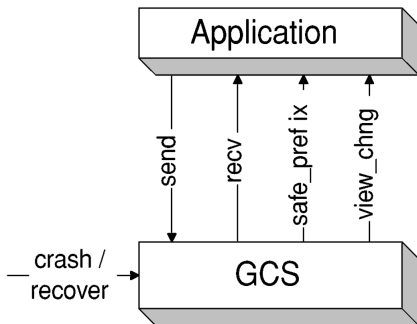
$\text{viewof}(t_i) =$ The view of a event t_i (send, etc.) occurring at p


\equiv The view delivered to p in a view_chng event t_j , which precedes t_i

and such that no view_chng or crash events occur at p between t_j and t_i

- The view is \perp if there is no such t_j

1. $\text{safe_prefix}(p, m)$ is ignored.





3 Safety Properties of the Group Membership Service

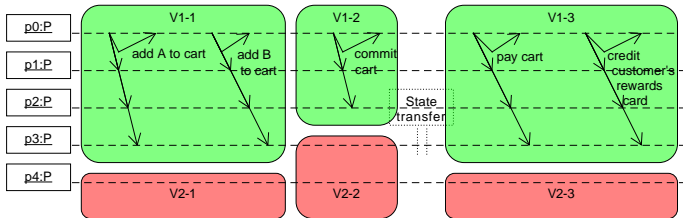
- 3.1 Basic Properties
- 3.2 Primary Component GCS
- 3.3 Partitionable GCS

3.1 Basic Properties

- Two preliminary properties
 - After a crash event, a recover event
 - Messages are unique, i.e. have unique identifiers
- Basic Properties
 - Self inclusion: If p installs view V , then p is a member of V
 - Local Monotonicity of view identifiers, e.g. do not install the same view twice
 - Initial view event: every send, recv occurs within some view

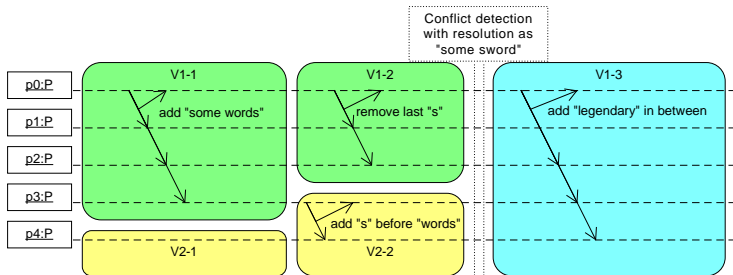
3.2 Primary Component GCS

- Primary Component: Views installed are totally ordered
 - Two consecutive views (V' , V'') intersect: $\exists p$ that survives from V' to V''
 - p conveys info. about msg exchanges from $V'.members$ to $V''.members$
 - Interesting for globally consistent shared state, e.g. state machine replication
 - Avoid inconsistencies: Only members of the primary view write data
 - Members of nonprimary views may access the data for reading purposes



3.3 Partitionable GCS

- Partitionable: Views are partially ordered
 - Multiple disjoint views exist concurrently
- Exemples appli.: resource alloc., load balancing, collaborative computing



4 Safety Properties of the Multicast Service

- 4.1 Basic Properties
- 4.2 Virtual Synchrony
- 4.3 Sending View / Same View Delivery
- 4.4 View-aware Ordering Properties
- 4.5 The case of Total ordering

- More in the article on order constraints for messages of different types, and on order constraints for multiple groups

4.1 Basic Properties

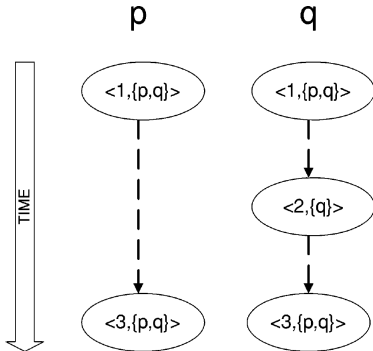
- Delivery integrity: For each `recv` there is a preceding `send` of the same message
- No duplication

4.2 Virtual Synchrony I

- Two processes that participate in the same two consecutive views deliver the same set of messages in the former
 - If p and q install V in V' , then any message received by p in V' is also received by q in V'
- Especially useful state machine replication
 - Applications change their state when they receive application messages
 - To maintain state consistency, messages are disseminated using totally ordered multicast
 - In case of network partitioning, disconnected replica may diverge
 - When “reconnecting”, perform a state transfer

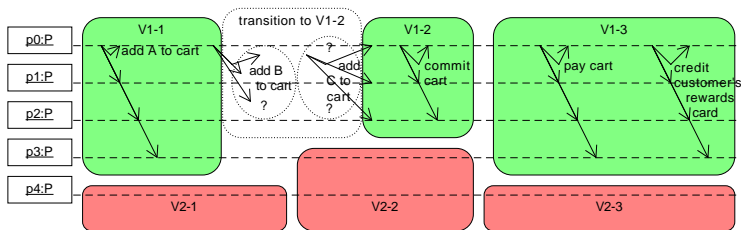
4.2 Virtual Synchrony II

- Virtual synchrony \implies avoid state transfer among processes that “continue together”
- Whenever p installs view “3”:
 - p should determine the set T of processes in “3”.members that were also in view “1” and have proceeded directly from 1 to 3
 - Directly: p installed view 1 and did not install any view after view 1 and before view 3
 - Process q is not in T



4.3 Sending View / Same View Delivery

- Sending View Delivery: If p receives message m in V , and q (possibly $p = q$) sends m in V' , then $V = V'$
- Same View Delivery: If processes p and q both receive message m , it is in the same view



In Sending View Delivery, "B is added" in view V_{1-1}

In Same View Delivery, "B is added" either in view V_{1-1} or in view V_{1-2}

Multicast "add C to card" is allowed only in Same View Delivery

4.3.1 Sending View Delivery's Pros and Cons

- Applications that exploit sending view delivery are called “view-aware”
- Pros:
 - Enable the Virtual Synchrony property
 - Minimize the amount of context information that needs to be sent with each message
 - E.g., state transfer messages sent when new views are installed:
No tagging of state transfer messages with the view in which it was sent
 - E.g., applications that send vectors of data corresponding to view members:
The i th entry in the vector = the i th member in the current view
- Cons:
 - Imply blocking sending of messages during a view change, i.e. a “flush”
 - In the absence of blocking, satisfaction of the sending view delivery without discarding messages from correct processes implies violation of the virtual synchrony property

4.4 View-aware Ordering Properties

- Some ordering properties are view-aware properties
 - For instance, FIFO prohibits gaps in the FIFO order only within a single view
 - Prohibiting gaps across views would require the GCS to log messages and retransmit them to new processes at view changes
 - GCSs generally do not log messages; thus, to be implemented atop GCSs
- Reliable FIFO: If p sends message m before message m' in the same view V , then any q that receives m' receives m before m'
- Reliable Causal: If m causally precedes m' , and both are sent in the same view, then any q that receives m' receives m before m'

4.5 The case of Total ordering

- Every message has got a timestamp
- Strong Total Order: There is a timestamp function f such that messages are received at all the processes in an order consistent with f
 - Requirement for one universal timestamp function
 - Interesting for consistent state replication, e.g. for state machine replication
- Reliable Total Order: There exists a timestamp function f such that if q receives m' , m and m' were sent in the same view, and $f(m) < f(m')$, then q receives m before m'
- Weak Total Order²: For every pair of views V and V' , there is a timestamp function f so that every process that installs V in V' receives messages in V' in an order consistent with f
 - Possibly different timestamp functions for each pair of views V and V'
 - Interesting for state replication with a reconciliation procedure when partitions merge, e.g. collaborative computing

2. We ignore here the specific situation of the “last view”

5 Conclusion I

■ Concepts and properties

- Active replication, state machine replication, collaborative computing
- Group membership, multicast, view, primary component Vs. partitionable
- Virtual synchrony, sending view delivery, same view delivery, safe messages
- View-aware ordering properties, strong and weak total order

■ More in the article

- Weak Virtual Synchrony, and optimistic Virtual Synchrony
- Order constraints for messages of different types, and order constraints for multiple groups
- Liveness properties

5 Conclusion II

- For application using the state machine replication approach, usually:
 - Primary component GCS
 - Virtual synchrony
 - Sending view delivery
 - Strong total order
- For applications such as collaborative computing/editing, usually:
 - Partionnable GCS
 - Same view delivery
 - Weak total order