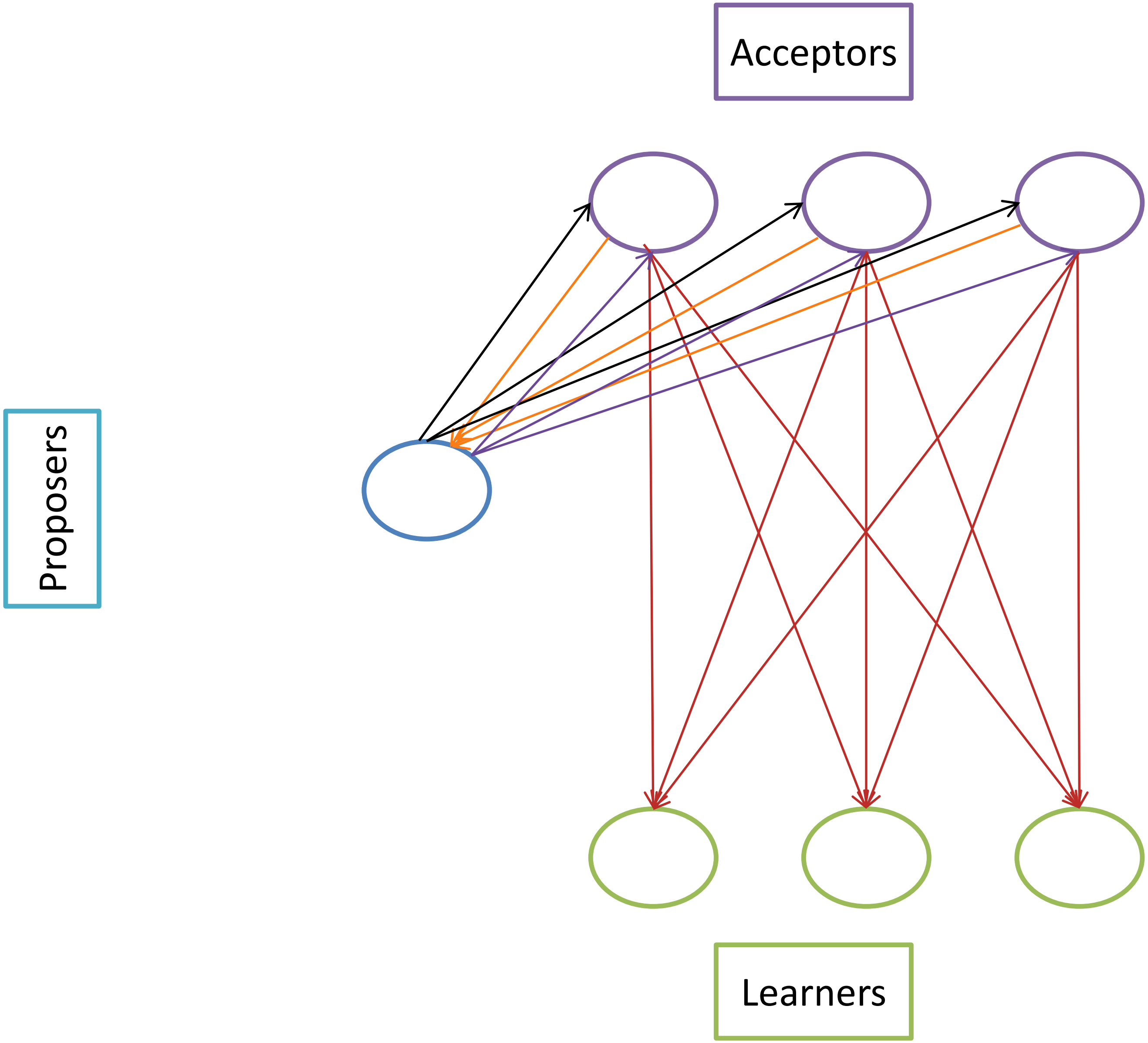
Distributed Consensus- Part 6: Paxos Implementation and Executions

Instructor: Prasun Dewan (FB 150, dewan@unc.edu)

Basic (Non Centralized) Paxos

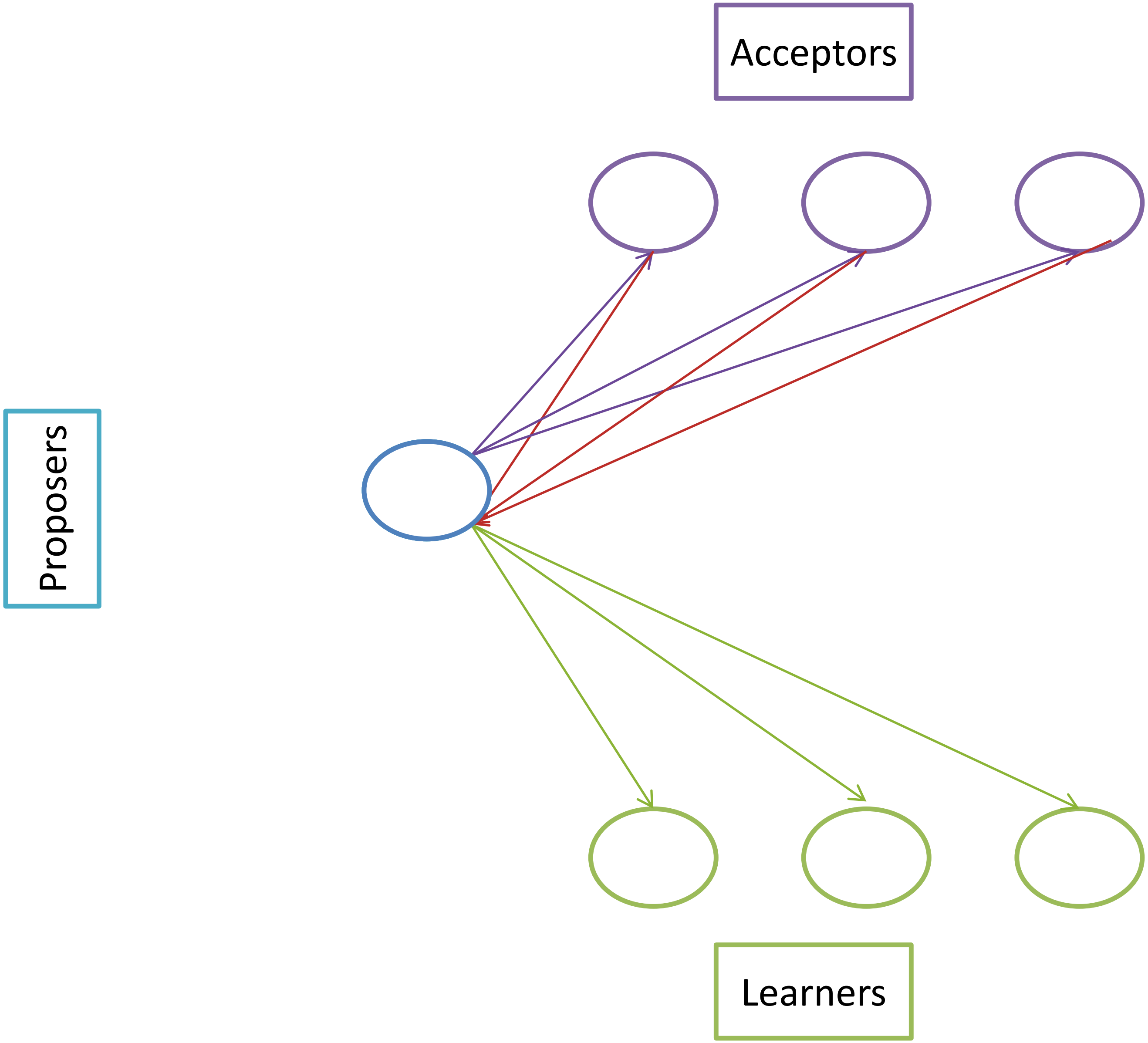


Acceptors

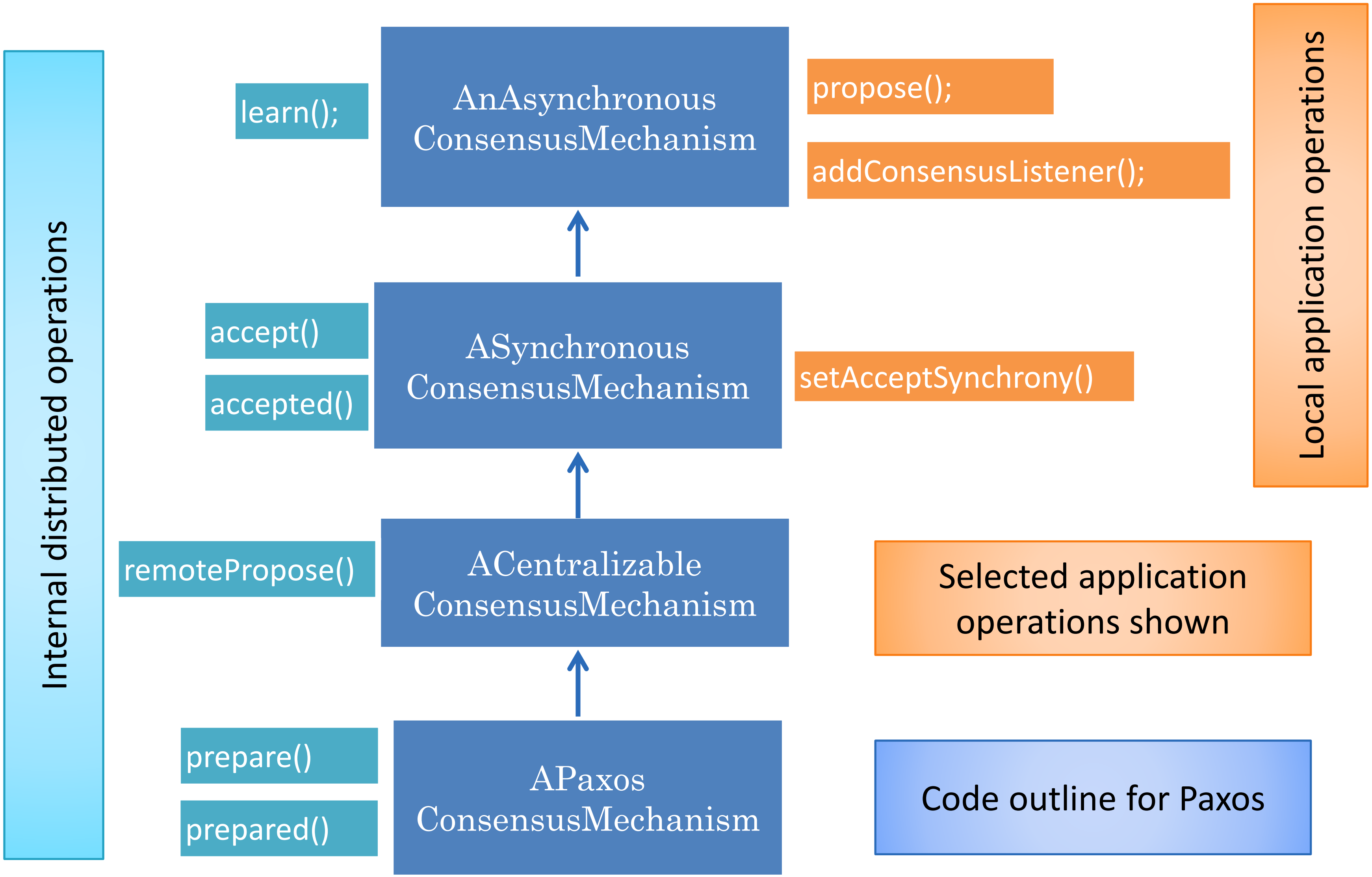
Proposers

Learners

(Central) Synchronous



Consensus Mechanism Hierarchy



Internal distributed operations

learn();

AnAsynchronous ConsensusMechanism

propose();

addConsensusListener();

Local application operations

accept()

accepted()

ASynchronous ConsensusMechanism

setAcceptSynchrony()

remotePropose()

ACentralizable ConsensusMechanism

Selected application operations shown

prepare()

prepared()

APaxos ConsensusMechanism

Code outline for Paxos

When to Not Use Paxos

```
protected boolean isNotPaxos() {
  return isNonAtomic() || isCentralizedPropose());
}
```

Propose First Phase

```
protected void localPropose(float aProposalNumber,
StateType aProposal) {
  if (isNotPaxos()) {
    super.localPropose(aProposalNumber, aProposal);
  } else {
    startPreparePhase(aProposalNumber, aProposal);
  }
}}
```

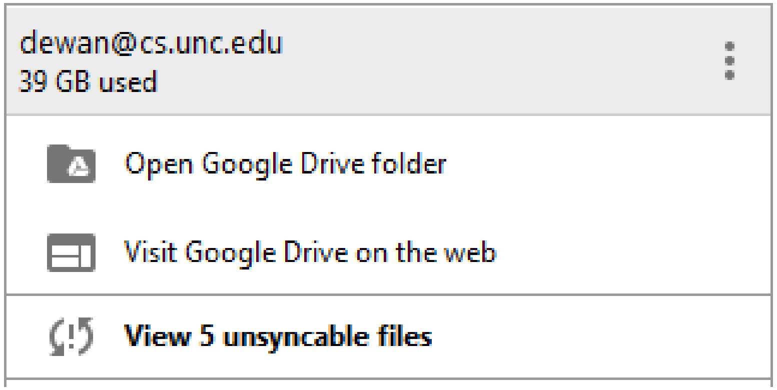
```
protected void startPreparePhase(float aProposalNumber,
StateType aProposal) {
  recordAndSendPrepareRequest(aProposalNumber, aProposal);
}
```

Prepare Query

```
public void prepare(float aProposalNumber, StateType aProposal) {
  float aPreparedOrAcceptedProposalNumber =
        maxProposalNumberSentInSuccessfulAcceptedNotification ;
  StateType anAcceptedState = null;
  if (aPreparedOrAcceptedProposalNumber != -1) {
    anAcceptedState = proposal(aPreparedOrAcceptedProposalNumber);
  } else if (sendPrepardNumberIfNoAccept()) {
    PreparedOrAcceptedProposalNumber =
        maxProposalNumberReceivedInPrepareOrAcceptRequest;
  }
  prepare(aPreparedOrAcceptedProposalNumber,
        anAcceptedState, aProposalNumber, aProposal,
        checkPrepareRequest(aProposalNumber, aProposal));
}
```

Each acceptor calculates and sends back ***last accepted value and its proposal number (or optionally last seen proposal number if no acceptance so far)*** and updates last seen proposal number

State-based Rejection



dewan@cs.unc.edu
39 GB used

Open Google Drive folder

Visit Google Drive on the web

View 5 unsyncable files

State-based rejection should be done as early as possible, so in prepare rather than accept phase and makes the safety condition in Lamport's algorithm stronger

```
protected synchronized ProposalFeedbackKind checkPrepareRequest(float
                 aProposalNumber, StateType aProposal ) {
  return  isPrepareConcurrencyConflict(aProposalNumber, aProposal)?
                 ProposalFeedbackKind.CONCURRENCY_CONFLICT:
                 checkWithVetoer(aProposalNumber, aProposal);
}
```

Single site cannot veto if majority used, rejection equal to not responding, which means majority wins if its response not considered

Prepare Concurrency-based Rejection

```
protected boolean isPrepareConcurrencyConflict (float aProposalNumber,
                 StateType aState )   {
  return
        maxProposalNumberReceivedInPrepareOrAcceptRequest >
        aProposalNumber;
}
```

Preparer abandons proposal if it learns about a higher number proposal

Prepare Query

```
public void prepare(float aProposalNumber, StateType aProposal) {
  float aPreparedOrAcceptedProposalNumber =
        maxProposalNumberSentInSuccessfulAcceptedNotification ;
  StateType anAcceptedState = null;
  if (aPreparedOrAcceptedProposalNumber != -1) {
    anAcceptedState = proposal(aPreparedOrAcceptedProposalNumber);
  } else if (sendPrepardNumberIfNoAccept()) {
    PreparedOrAcceptedProposalNumber =
        maxProposalNumberReceivedInPrepareOrAcceptRequest;
  }
  prepare(aPreparedOrAcceptedProposalNumber,
        anAcceptedState, aProposalNumber, aProposal,
        checkPrepareRequest(aProposalNumber, aProposal));
}
```

Each acceptor calculates and sends back ***last accepted value and its proposal number (or optionally last seen proposal number if no acceptance so far)*** and updates last seen proposal number

Helper Prepare

```
protected void prepare(float aLastPreparedOrAcceptedProposalNumber, StateType
         aLastAcceptedProposal, float aPreparedProposalNumber,
         StateType aProposal,   ProposalFeedbackKind aFeedbackKind) {
  recordReceivedPrepareRequest(aPreparedProposalNumber, aProposal);
  if (
         // we accepted this proposal before preparing for it
         aPreparedProposalNumber == aLastPreparedOrAcceptedProposalNumber
         // peparer has started the accept phase
         || !isPending(aPreparedProposalNumber)) {
    return;
  }
  if (!isSuccess(aFeedbackKind)) {
     processPrepareRejection(aLastPreparedOrAcceptedProposalNumber,
         aLastAcceptedProposal, aPreparedProposalNumber, aFeedbackKind);
  } else {
    recordAndSendPrepareResponse(aLastPreparedOrAcceptedProposalNumber,
         aLastAcceptedProposal, aPreparedProposalNumber, aFeedbackKind);
  }
}
```

Each acceptor **calculates** and ***sends back*** sends back last accepted value and its proposal number (or optionally last seen proposal number if no acceptance so far) ***and updates last seen proposal number***

Prepared

```
public void prepared(float aPreparedOrAcceptedProposalNumber,
StateType anAcceptedProposal, float aPreparedProposalNumber,
ProposalFeedbackKind aFeedbackKind) {
  recordReceivedPreparedNotification(aPreparedOrAcceptedProposalNumber,
         anAcceptedProposal, aPreparedProposalNumber, aFeedbackKind);
  if (!isPending(aPreparedProposalNumber)
        || isPreparePhaseOver(aPreparedProposalNumber)) {
   return;
  }
  if (aFeedbackKind == ProposalFeedbackKind.CONCURRENCY_CONFLICT) {
    newProposalState( aPreparedProposalNumber,
         proposal(aPreparedProposalNumber),
         toProposalState(
             aPreparedProposalNumber,anAcceptedProposal, aFeedbackKind));
     return;
  }
  aggregatePreparedNotification(aPreparedOrAcceptedProposalNumber,
    anAcceptedProposal, aPreparedProposalNumber, aFeedbackKind);
}
```

Preparer abandons proposal if it learns about a higher number proposal

Prepare Aggregation

```
protected void aggregatePreparedNotification(
float anAcceptedProposalNumber, StateType anAcceptedProposal,
float aPreparedProposalNumber, ProposalFeedbackKind aFeedbackKind) {
  Boolean isSufficientPreparers = sufficientPeparers(
        getPrepareSynchrony(), aPreparedProposalNumber);
  if (isSufficientPreparers == null)
    return;
  setPreparePhaseOver(aPreparedProposalNumber);
  if (isSufficientPreparers) {
    startAcceptPhase(aPreparedProposalNumber,
        preparedProposal(aPreparedProposalNumber));
  } else {
    newProposalState( aPreparedProposalNumber,
        proposal(aPreparedProposalNumber),
        ProposalState.PROPOSAL_AGGREGATE_DENIAL);
    return;
  }
}
```

Prepared Proposal

```
protected StateType preparedProposal(float aPreparedProposalNumber) {
  float aChosenProposalNumber =
        maxAcceptedProposalNumberReceivedInPreparedNotification <= 0 ?
                aPreparedProposalNumber
                : maxAcceptedProposalNumberReceivedInPreparedNotification;
  return proposal(aChosenProposalNumber);
}
```

Proposer sends its proposal and value if majority acceptors have not yet accepted any value

Proposer (re) proposes with highest accept proposal number as its own value (which may also be a majority value in majority acceptors)

Proposal Phase 2

```
protected void startAcceptPhase(float aProposalNumber, StateType aProposal)
{
  super.startAcceptPhase(aProposalNumber, aProposal);
}
```

Accept Check

```
protected synchronized ProposalFeedbackKind checkAcceptRequest(float
aProposalNumber, StateType aProposal ) {
  if (isNotPaxos()) {
    return super.checkAcceptRequest(aProposalNumber, aProposal);
  }
  return (isAcceptConcurrencyConflict(aProposalNumber, aProposal))?
    ProposalFeedbackKind.CONCURRENCY_CONFLICT:
    ProposalFeedbackKind.SUCCESS;
 }
```

No application-specific check in this phase under Paxos

Prepare-Accept Integration

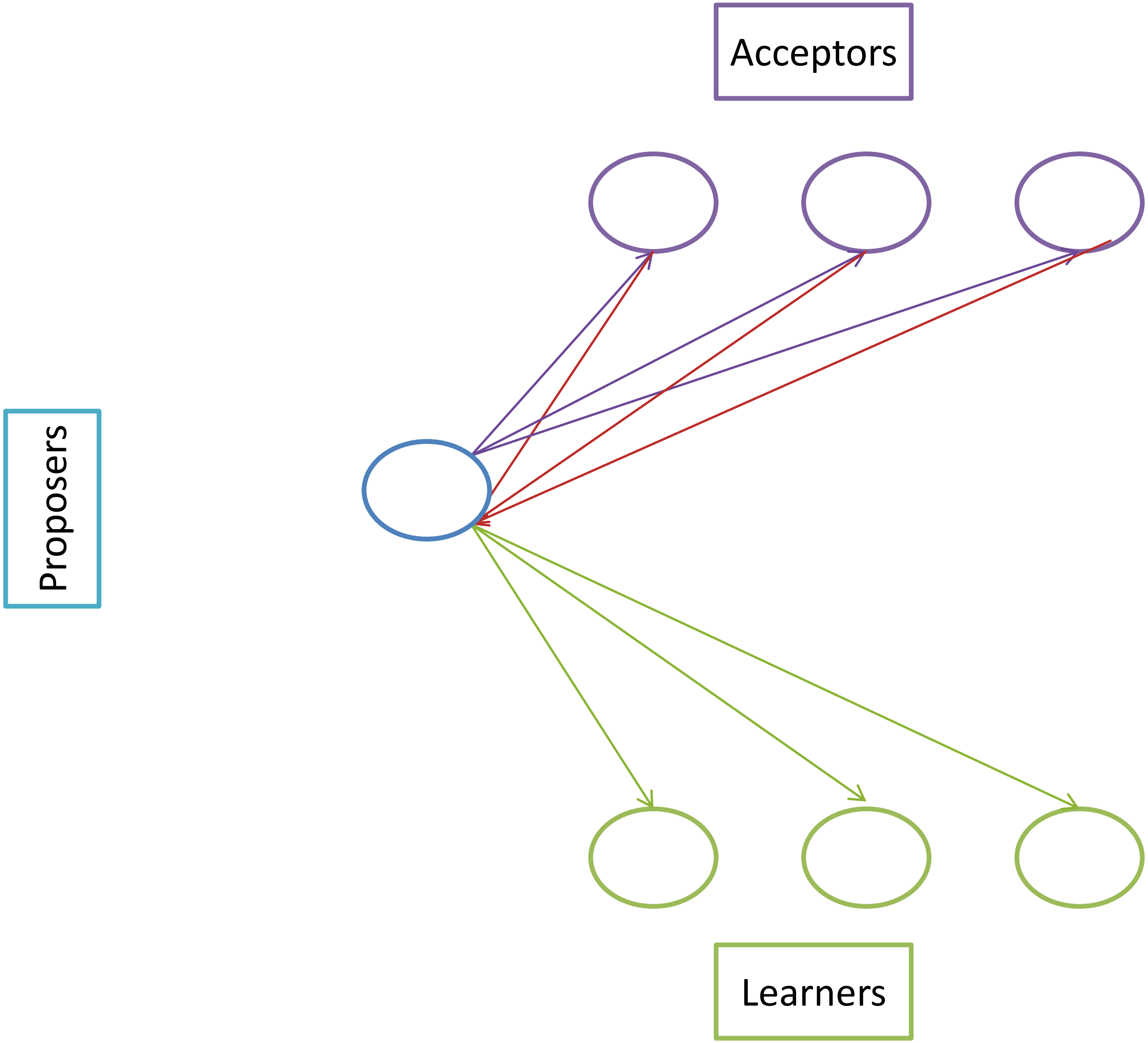
```
protected boolean isAcceptConcurrencyConflict (float aProposalNumber,
StateType aState )  {
   return isPrepareConcurrencyConflict(aProposalNumber, aState);
}
```

```
protected void recordReceivedAcceptRequest(float aProposalNumber,
StateType aProposal) {
  super.recordReceivedAcceptRequest(aProposalNumber, aProposal);
  maxProposalNumberReceivedInPrepareOrAcceptRequest = Math.max(
     maxProposalNumberReceivedInPrepareOrAcceptRequest, aProposalNumber);
}
```

Rejection reasons same for prepare and accept

Last seen proposal number updated by both

Synchronous Accept Phase

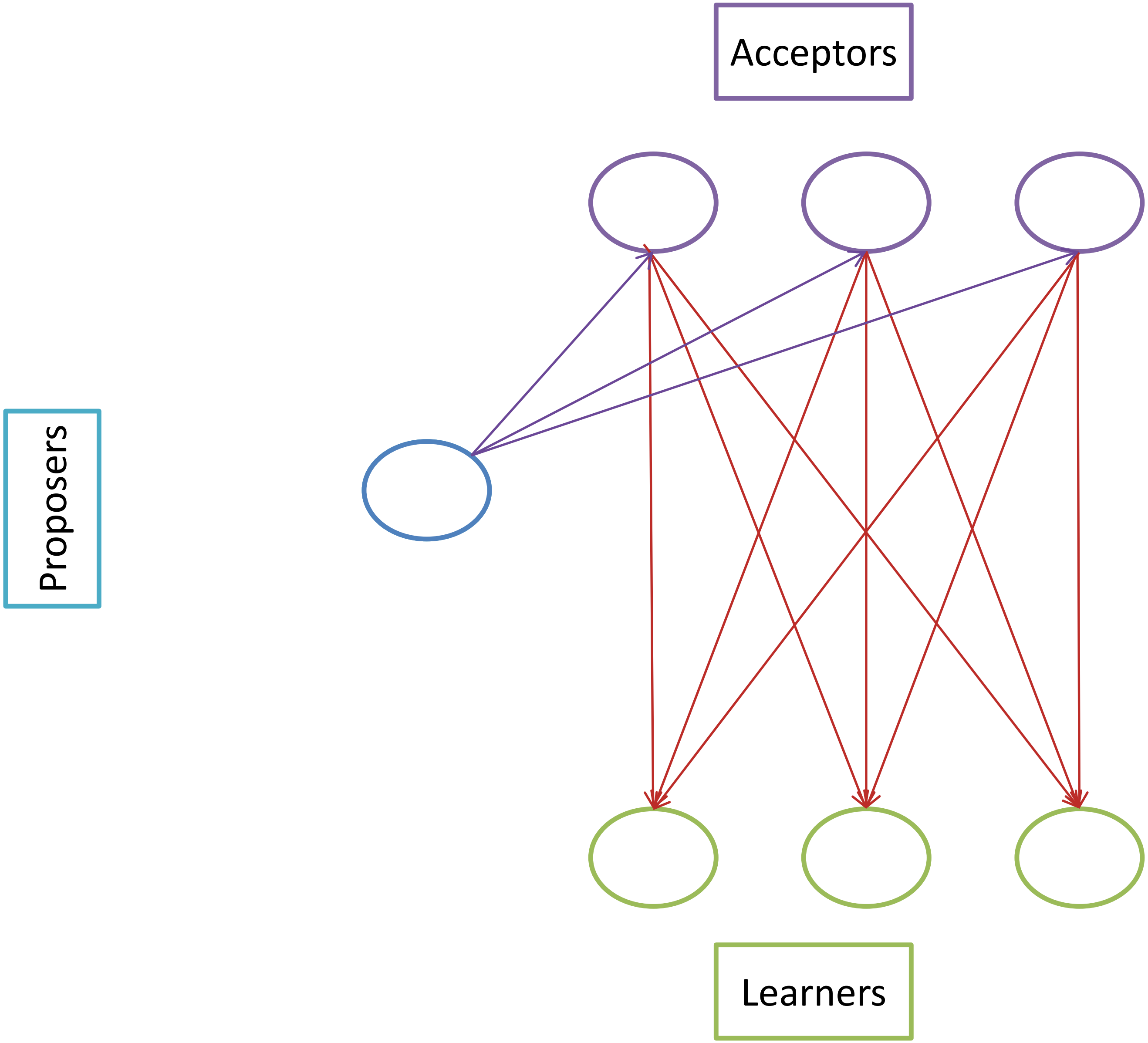


Acceptors

Proposers

Learners

Basic (Non Centralized) Paxos Accept Phase



Acceptors

Proposers

Learners

Do not Broadcast Learned Information

Super class code

```
protected void sendLearnNotification(float aProposalNumber,
StateType aProposal, ProposalFeedbackKind anAgreement) {
  localLearn(aProposalNumber, aProposal, anAgreement);
  sendLearnNotificationToOthers(aProposalNumber, aProposal, anAgreement);
}
```

Paxos-specific code

```
protected void sendLearnNotificationToOthers(float aProposalNumber,
StateType aProposal, ProposalFeedbackKind anAgreement) {
  if (isNotPaxos()) {
   super.sendLearnNotificationToOthers(aProposalNumber,
         aProposal, anAgreement);
  }
}
```

Learned values are not broadcast in Basic Paxos

Broadcast Accepted Notification

```
protected void sendAcceptedNotification(float aProposalNumber,
StateType aProposal, ProposalFeedbackKind aFeedbackKind) {
  if (isNotPaxos()) {
    super.sendAcceptedNotification(aProposalNumber, aProposal,
         aFeedbackKind);
    return;
  }
  sendAcceptedNotificationToLearners(aProposalNumber, aProposal,
         aFeedbackKind);
}
```

Accepted notifications sent to everyone

Example Scenarios

1 2 3

1

```
proposeMeaning(MEANING_1);
```

3

```
proposeMeaning(MEANING_2);
```

Paxos Algorithm Properties

Proposer sends its proposal and value if majority acceptors have not yet accepted any value

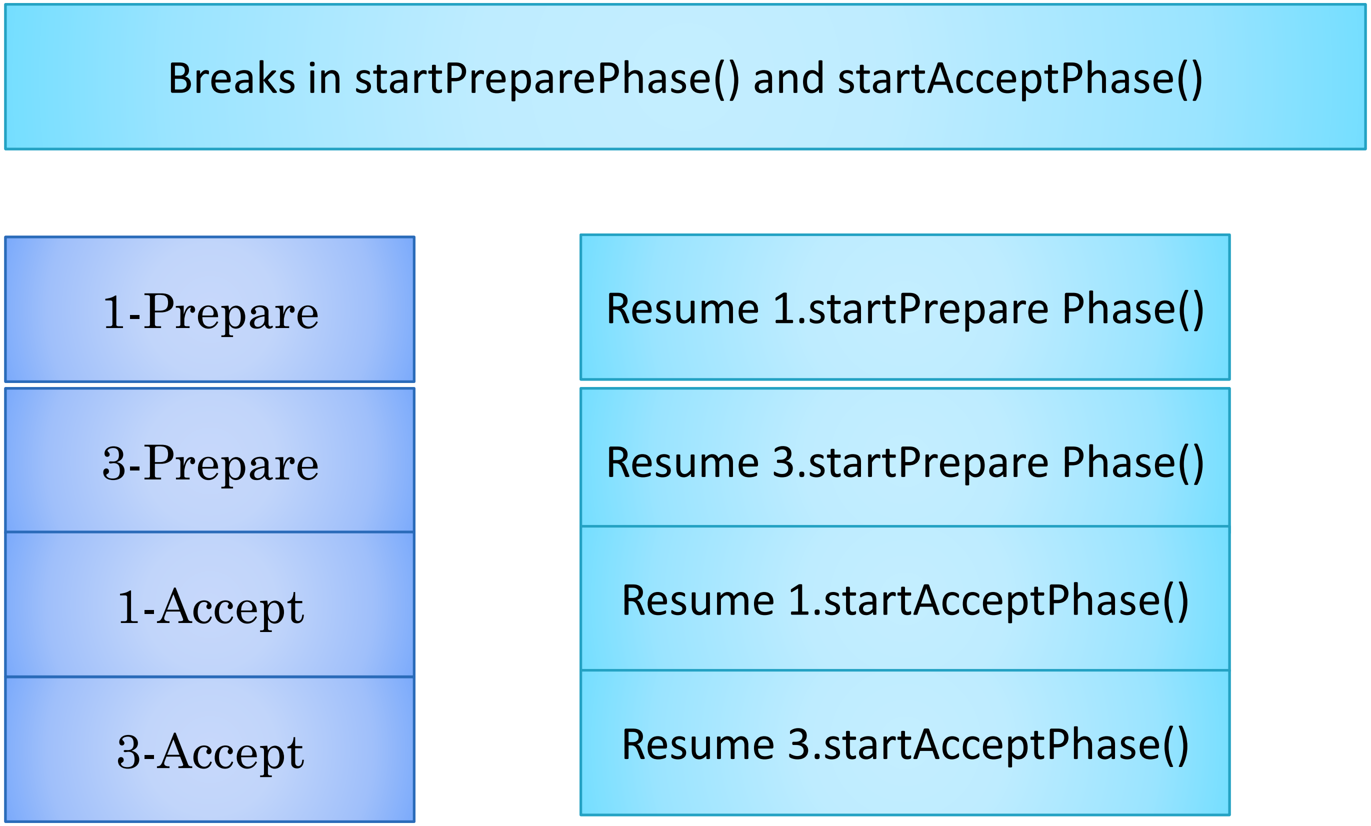
Each acceptor accepts a proposal if its proposal number is higher than what it has seen so far.

Case 1 Properties

1 and 3 finds no previous acceptance and 1 finds no previous prepare

1’s accept is rejected by majority, 3’s accept goes through

Case 1: In-Order Prepares Before Accepts



Breaks in startPreparePhase() and startAcceptPhase()

1-Prepare

3-Prepare

1-Accept

3-Accept

Resume 1.startPrepare Phase()

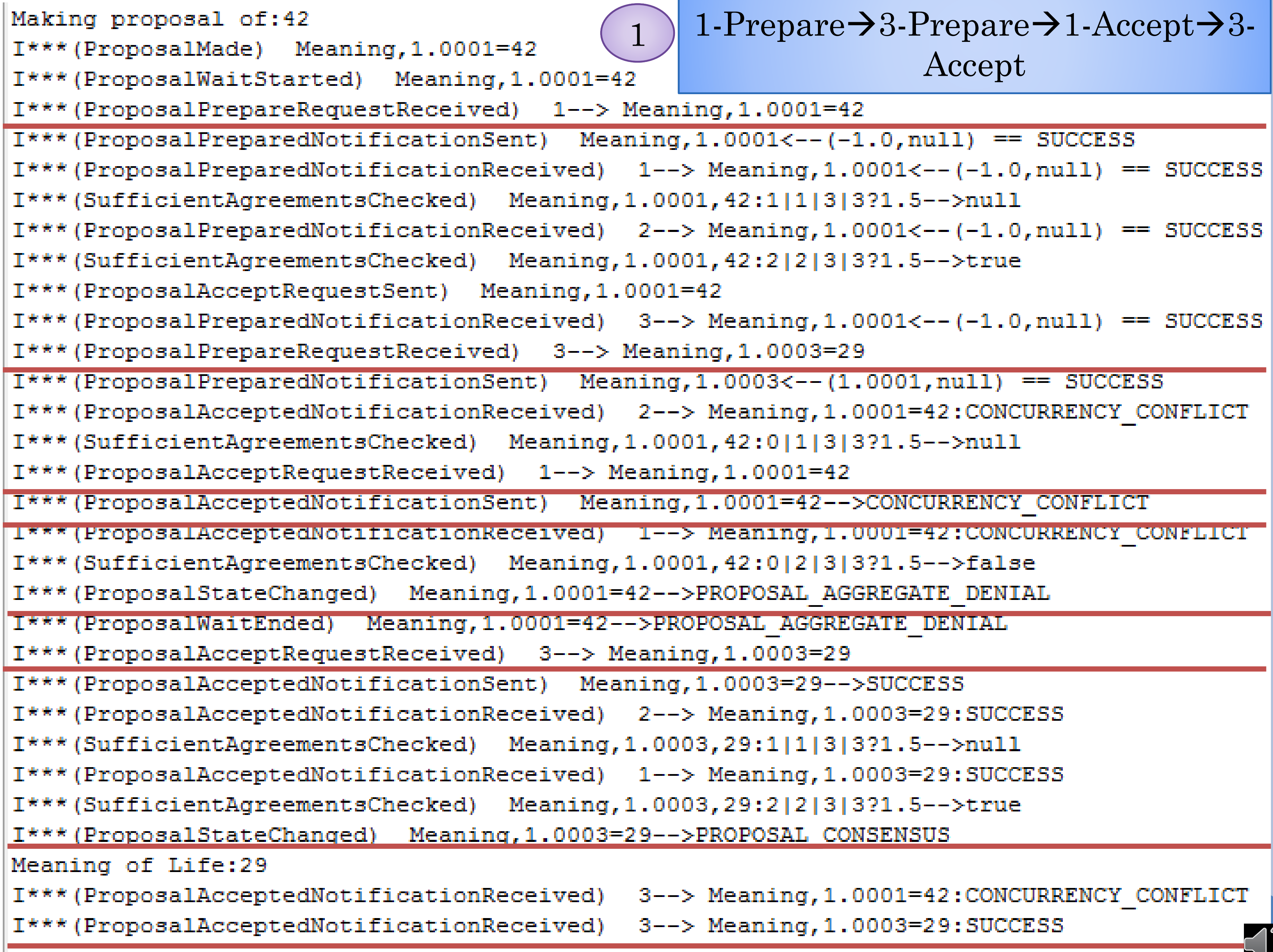
Resume 3.startPrepare Phase()

Resume 1.startAcceptPhase()

Resume 3.startAcceptPhase()

1-Prepare→3-Prepare→1-Accept→3-Accept

1



```
Making proposal of:42
I***(ProposalMade)  Meaning,1.0001=42
I***(ProposalWaitStarted)  Meaning,1.0001=42
I***(ProposalPrepareRequestReceived)  1--> Meaning,1.0001=42
I***(ProposalPreparedNotificationSent)  Meaning,1.0001<--(-1.0,null) == SUCCESS
I***(ProposalPreparedNotificationReceived)  1--> Meaning,1.0001<--(-1.0,null) == SUCCESS
I***(SufficientAgreementsChecked)  Meaning,1.0001,42:1|1|3|3?1.5-->null
I***(ProposalPreparedNotificationReceived)  2--> Meaning,1.0001<--(-1.0,null) == SUCCESS
I***(SufficientAgreementsChecked)  Meaning,1.0001,42:2|2|3|3?1.5-->true
I***(ProposalAcceptRequestSent)  Meaning,1.0001=42
I***(ProposalPreparedNotificationReceived)  3--> Meaning,1.0001<--(-1.0,null) == SUCCESS
I***(ProposalPrepareRequestReceived)  3--> Meaning,1.0003=29
I***(ProposalPreparedNotificationSent)  Meaning,1.0003<--(1.0001,null) == SUCCESS
I***(ProposalAcceptedNotificationReceived)  2--> Meaning,1.0001=42:CONCURRENCY_CONFLICT
I***(SufficientAgreementsChecked)  Meaning,1.0001,42:0|1|3|3?1.5-->null
I***(ProposalAcceptRequestReceived)  1--> Meaning,1.0001=42
I***(ProposalAcceptedNotificationSent)  Meaning,1.0001=42-->CONCURRENCY_CONFLICT
I***(ProposalAcceptedNotificationReceived)  1--> Meaning,1.0001=42:CONCURRENCY_CONFLICT
I***(SufficientAgreementsChecked)  Meaning,1.0001,42:0|2|3|3?1.5-->false
I***(ProposalStateChanged)  Meaning,1.0001=42-->PROPOSAL_AGGREGATE_DENIAL
I***(ProposalWaitEnded)  Meaning,1.0001=42-->PROPOSAL_AGGREGATE_DENIAL
I***(ProposalAcceptRequestReceived)  3--> Meaning,1.0003=29
I***(ProposalAcceptedNotificationSent)  Meaning,1.0003=29-->SUCCESS
I***(ProposalAcceptedNotificationReceived)  2--> Meaning,1.0003=29:SUCCESS
I***(SufficientAgreementsChecked)  Meaning,1.0003,29:1|1|3|3?1.5-->null
I***(ProposalAcceptedNotificationReceived)  1--> Meaning,1.0003=29:SUCCESS
I***(SufficientAgreementsChecked)  Meaning,1.0003,29:2|2|3|3?1.5-->true
I***(ProposalStateChanged)  Meaning,1.0003=29-->PROPOSAL_CONSENSUS
Meaning of Life:29
I***(ProposalAcceptedNotificationReceived)  3--> Meaning,1.0001=42:CONCURRENCY_CONFLICT
I***(ProposalAcceptedNotificationReceived)  3--> Meaning,1.0003=29:SUCCESS
```

Paxos Algorithm Properties

Preparer abandons proposal if it learns about a higher number proposal

Case 2 Properties

1 finds a higher proposal in prepare phase

Case 2: Reverse-Order Prepares Before Accepts

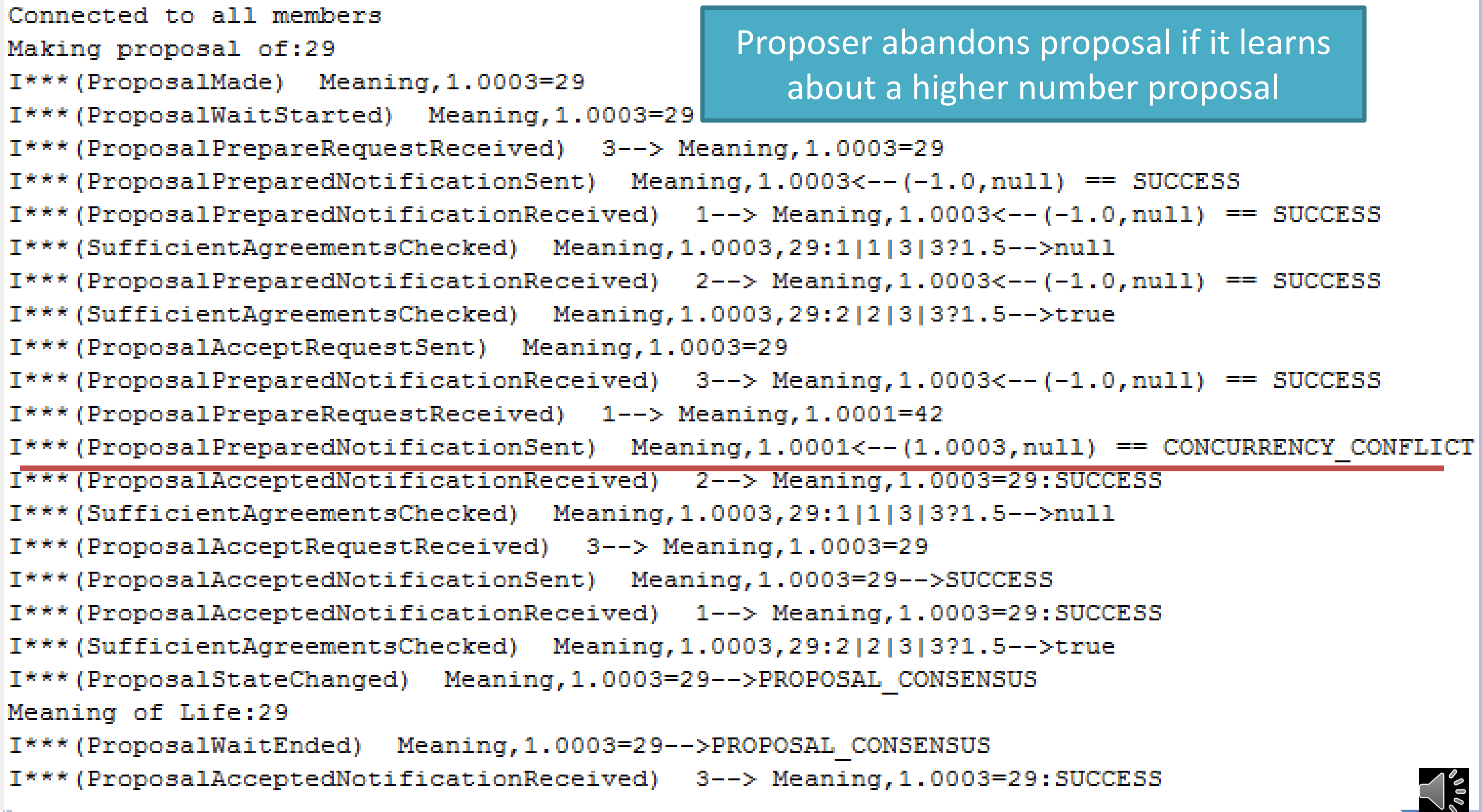
Breaks in startPreparePhase() and startAcceptPhase() in 1 and 3

3-Prepare	Resume 3.startPrepare Phase()
1-Prepare	Resume 1.startPrepare Phase()
1-Accept	Resume 1.startAcceptPhase()
3-Accept	Resume 3.startAcceptPhase()

Case 2: Abandoning Lower Proposal in Prepare Phase

3

1-Prepare→3-Prepare→3-Accept



Proposer abandons proposal if it learns about a higher number proposal

```
Connected to all members
Making proposal of:29
I***(ProposalMade)  Meaning,1.0003=29
I***(ProposalWaitStarted)  Meaning,1.0003=29
I***(ProposalPrepareRequestReceived)  3--> Meaning,1.0003=29
I***(ProposalPreparedNotificationSent)  Meaning,1.0003<--(-1.0,null) == SUCCESS
I***(ProposalPreparedNotificationReceived)  1--> Meaning,1.0003<--(-1.0,null) == SUCCESS
I***(SufficientAgreementsChecked)  Meaning,1.0003,29:1|1|3|3?1.5-->null
I***(ProposalPreparedNotificationReceived)  2--> Meaning,1.0003<--(-1.0,null) == SUCCESS
I***(SufficientAgreementsChecked)  Meaning,1.0003,29:2|2|3|3?1.5-->true
I***(ProposalAcceptRequestSent)  Meaning,1.0003=29
I***(ProposalPreparedNotificationReceived)  3--> Meaning,1.0003<--(-1.0,null) == SUCCESS
I***(ProposalPrepareRequestReceived)  1--> Meaning,1.0001=42
I***(ProposalPreparedNotificationSent)  Meaning,1.0001<--(1.0003,null) == CONCURRENCY_CONFLICT
I***(ProposalAcceptedNotificationReceived)  2--> Meaning,1.0003=29:SUCCESS
I***(SufficientAgreementsChecked)  Meaning,1.0003,29:1|1|3|3?1.5-->null
I***(ProposalAcceptRequestReceived)  3--> Meaning,1.0003=29
I***(ProposalAcceptedNotificationSent)  Meaning,1.0003=29-->SUCCESS
I***(ProposalAcceptedNotificationReceived)  1--> Meaning,1.0003=29:SUCCESS
I***(SufficientAgreementsChecked)  Meaning,1.0003,29:2|2|3|3?1.5-->true
I***(ProposalStateChanged)  Meaning,1.0003=29-->PROPOSAL_CONSENSUS
Meaning of Life:29
I***(ProposalWaitEnded)  Meaning,1.0003=29-->PROPOSAL_CONSENSUS
I***(ProposalAcceptedNotificationReceived)  3--> Meaning,1.0003=29:SUCCESS
```

Paxos Algorithm Property

Proposer (re) proposes majority value learned from the prepare phase as its own value

Case 3 Property

3 finds majority acceptances from 1 in prepare phase

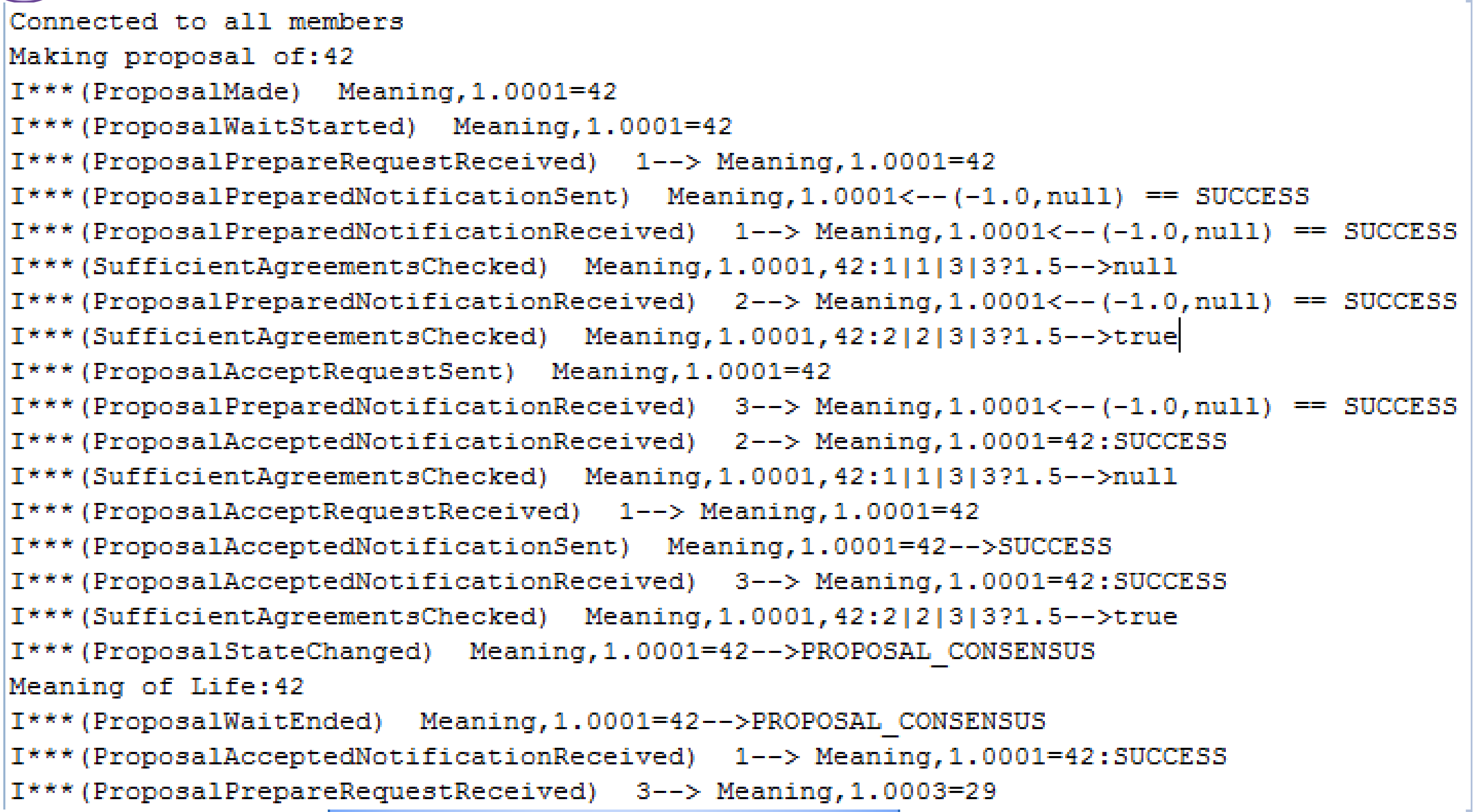
Case 3: 1 Before 3

Breaks in startPreparePhase() and startAcceptPhase() in 1 and 3

1-Prepare	Resume 1.startPrepare Phase()
1-Accept	Resume 1.startAcceptPhase()
3-Prepare	Resume 3.startPrepare Phase()
3-Accept	Resume 3.startAcceptPhase()

Case 3

1

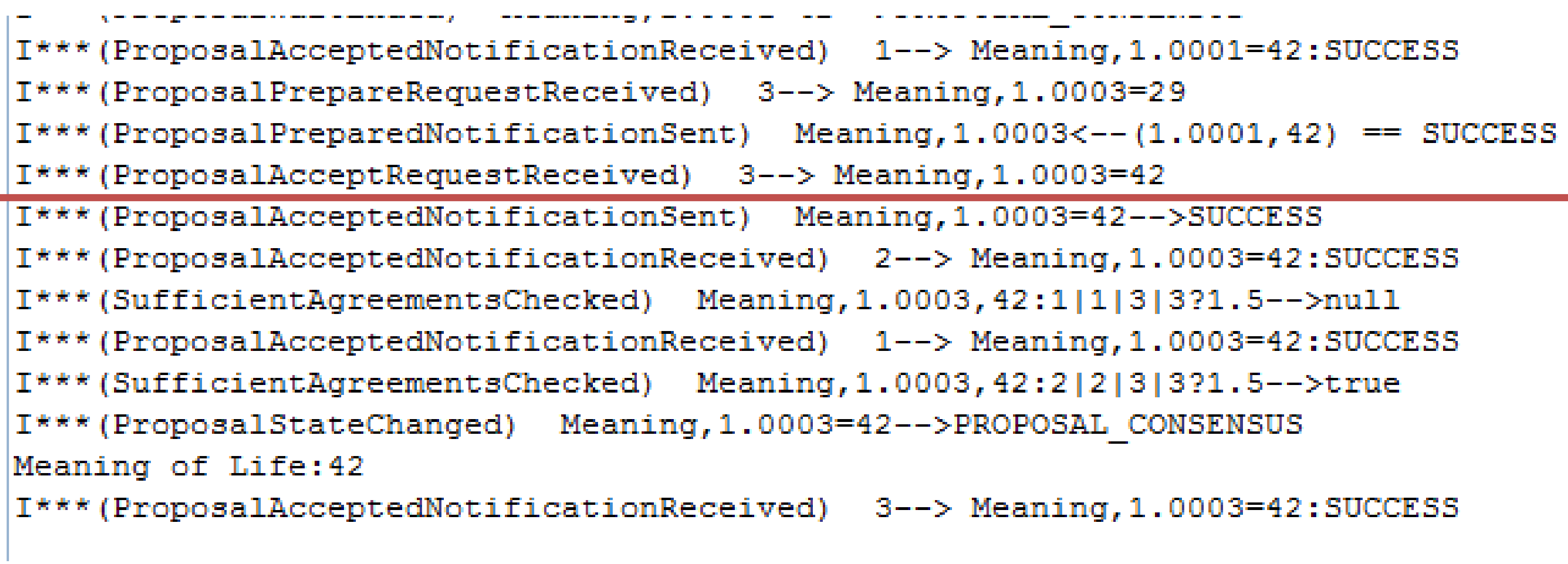


```
Connected to all members
Making proposal of:42
I***(ProposalMade)  Meaning,1.0001=42
I***(ProposalWaitStarted)  Meaning,1.0001=42
I***(ProposalPrepareRequestReceived)  1--> Meaning,1.0001=42
I***(ProposalPreparedNotificationSent)  Meaning,1.0001<--(-1.0,null) == SUCCESS
I***(ProposalPreparedNotificationReceived)  1--> Meaning,1.0001<--(-1.0,null) == SUCCESS
I***(SufficientAgreementsChecked)  Meaning,1.0001,42:1|1|3|3?1.5-->null
I***(ProposalPreparedNotificationReceived)  2--> Meaning,1.0001<--(-1.0,null) == SUCCESS
I***(SufficientAgreementsChecked)  Meaning,1.0001,42:2|2|3|3?1.5-->true
I***(ProposalAcceptRequestSent)  Meaning,1.0001=42
I***(ProposalPreparedNotificationReceived)  3--> Meaning,1.0001<--(-1.0,null) == SUCCESS
I***(ProposalAcceptedNotificationReceived)  2--> Meaning,1.0001=42:SUCCESS
I***(SufficientAgreementsChecked)  Meaning,1.0001,42:1|1|3|3?1.5-->null
I***(ProposalAcceptRequestReceived)  1--> Meaning,1.0001=42
I***(ProposalAcceptedNotificationSent)  Meaning,1.0001=42-->SUCCESS
I***(ProposalAcceptedNotificationReceived)  3--> Meaning,1.0001=42:SUCCESS
I***(SufficientAgreementsChecked)  Meaning,1.0001,42:2|2|3|3?1.5-->true
I***(ProposalStateChanged)  Meaning,1.0001=42-->PROPOSAL_CONSENSUS
Meaning of Life:42
I***(ProposalWaitEnded)  Meaning,1.0001=42-->PROPOSAL_CONSENSUS
I***(ProposalAcceptedNotificationReceived)  1--> Meaning,1.0001=42:SUCCESS
I***(ProposalPrepareRequestReceived)  3--> Meaning,1.0003=29
```

1-Prepare-1-Accept

Late Messages and Re-proposals

1



```
I***(ProposalAcceptedNotificationReceived)  1--> Meaning,1.0001=42:SUCCESS
I***(ProposalPrepareRequestReceived)  3--> Meaning,1.0003=29
I***(ProposalPreparedNotificationSent)  Meaning,1.0003<--(1.0001,42) == SUCCESS
I***(ProposalAcceptRequestReceived)  3--> Meaning,1.0003=42
I***(ProposalAcceptedNotificationSent)  Meaning,1.0003=42-->SUCCESS
I***(ProposalAcceptedNotificationReceived)  2--> Meaning,1.0003=42:SUCCESS
I***(SufficientAgreementsChecked)  Meaning,1.0003,42:1|1|3|3?1.5-->null
I***(ProposalAcceptedNotificationReceived)  1--> Meaning,1.0003=42:SUCCESS
I***(SufficientAgreementsChecked)  Meaning,1.0003,42:2|2|3|3?1.5-->true
I***(ProposalStateChanged)  Meaning,1.0003=42-->PROPOSAL_CONSENSUS
Meaning of Life:42
I***(ProposalAcceptedNotificationReceived)  3--> Meaning,1.0003=42:SUCCESS
```

3-Prepare-3-Accept

Why Re-propose with Majority?

Proposer (re) proposes majority value learned from the prepare phase as its own value

A prepare phase can prevent some nodes from accepting the previous (to be) majority value

If some of the nodes in current majority die, some learner nodes may not get consensus value even though majority of nodes are alive and can converge to a value

Want consensus value to propagate to all acceptors for fault tolerance

Case 4: Properties

A prepare phase prevents node 3 from accepting the previous majority value of 1’s proposal accepted by 1 and 2

If node 2 dies, node 3 will not get consensus value even though majority of nodes (1 and 3) are alive and can converge to a value

Want consensus value to propagate to all acceptors for fault tolerance

Break in startAcceptPhase() at start, sendAcceptedFrom2() before sending to 3, sendPrepareFrom3() before sending to 1,2

1-Prepare-*

3-Prepare-3

3 will not get 1’s proposal but 1 and 2 will

1-Accept-*

Resume 1.startAcceptPhase

1-Accepted-*

2-Accepted-1,2

3-Prepare-1,2

Resume 3.sendPrepareFrom3()

2’s accepted notification will not reach 3 but will reach 1 and 2

Kill 2

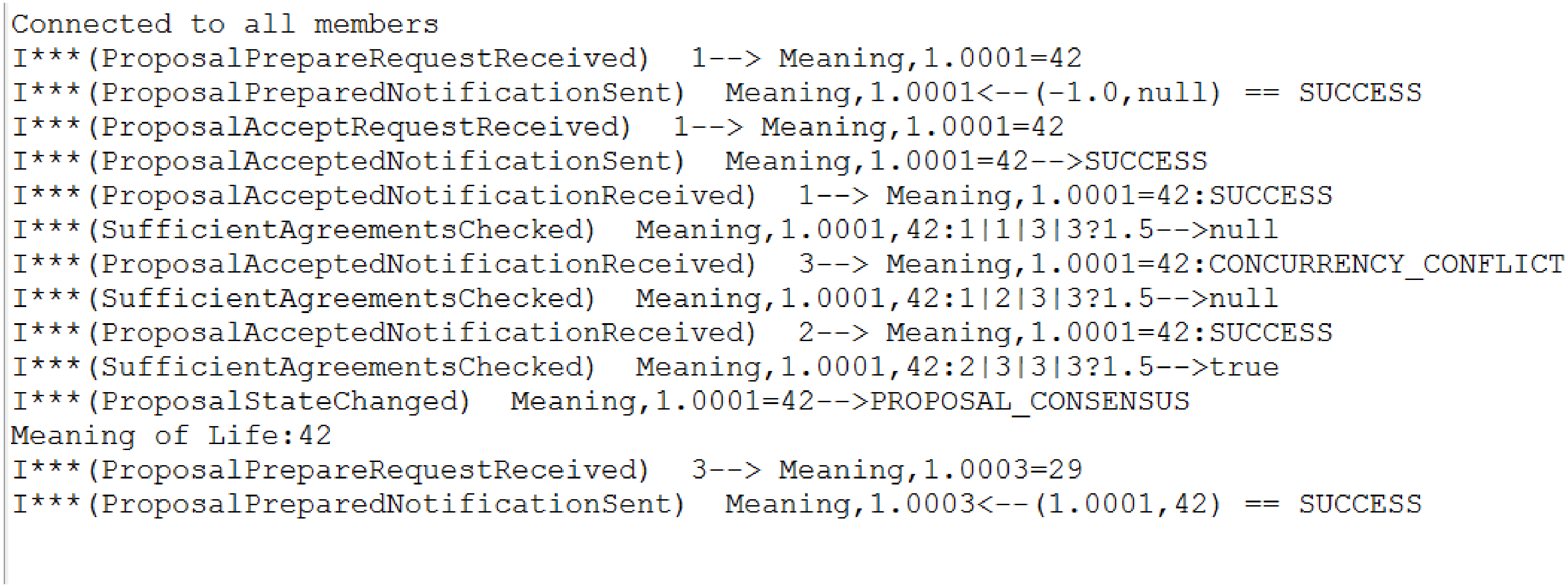
3-Accept-*

Resume Break 3. startAcceptPhase)

So 3 should re-propose

Case 4: 2 Learns Before Dying

2

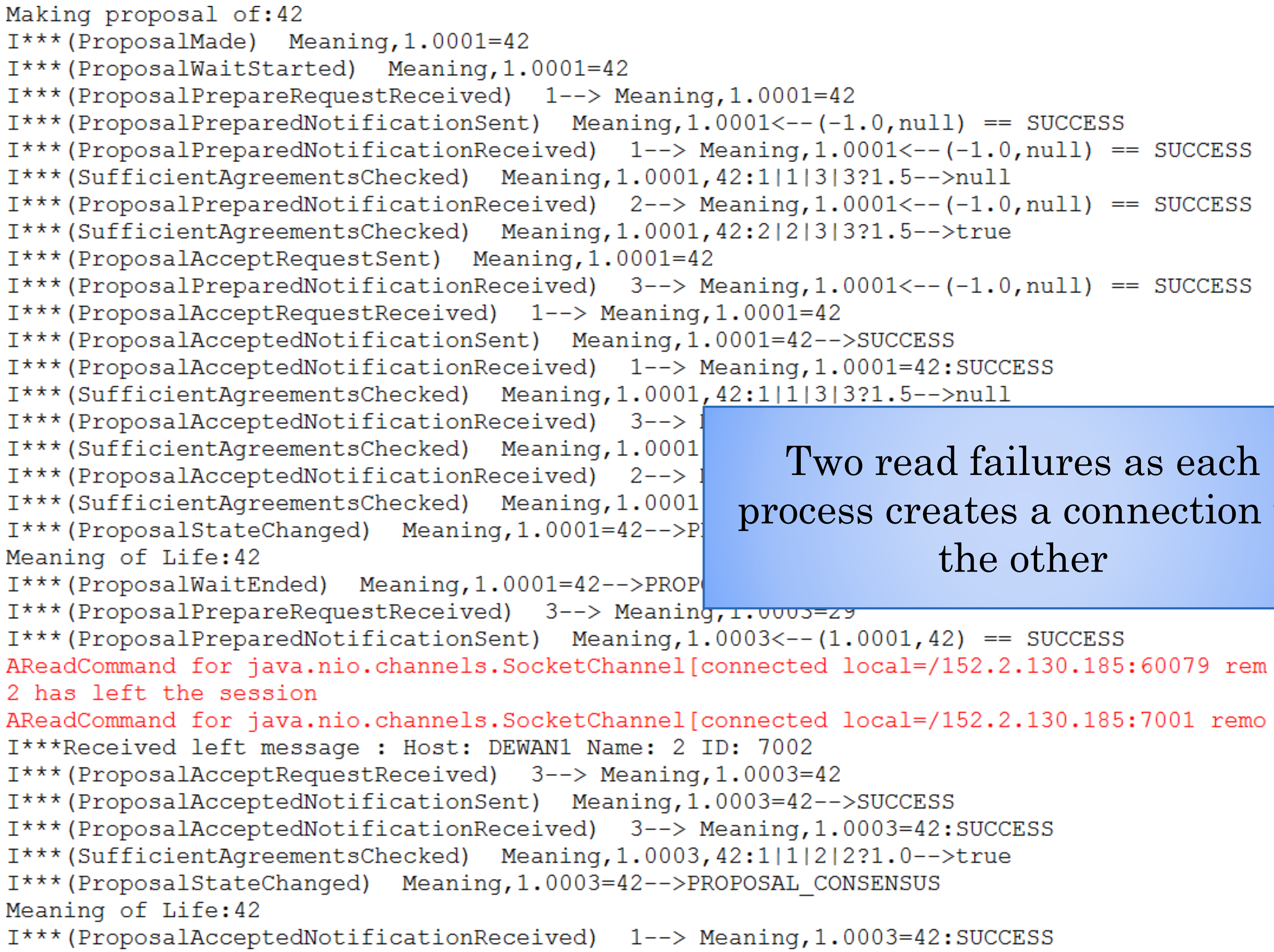


```
Connected to all members
I***(ProposalPrepareRequestReceived)  1--> Meaning,1.0001=42
I***(ProposalPreparedNotificationSent)  Meaning,1.0001<--(-1.0,null) == SUCCESS
I***(ProposalAcceptRequestReceived)  1--> Meaning,1.0001=42
I***(ProposalAcceptedNotificationSent)  Meaning,1.0001=42-->SUCCESS
I***(ProposalAcceptedNotificationReceived)  1--> Meaning,1.0001=42:SUCCESS
I***(SufficientAgreementsChecked)  Meaning,1.0001,42:1|1|3|3?1.5-->null
I***(ProposalAcceptedNotificationReceived)  3--> Meaning,1.0001=42:CONCURRENCY_CONFLICT
I***(SufficientAgreementsChecked)  Meaning,1.0001,42:1|2|3|3?1.5-->null
I***(ProposalAcceptedNotificationReceived)  2--> Meaning,1.0001=42:SUCCESS
I***(SufficientAgreementsChecked)  Meaning,1.0001,42:2|3|3|3?1.5-->true
I***(ProposalStateChanged)  Meaning,1.0001=42-->PROPOSAL_CONSENSUS
Meaning of Life:42
I***(ProposalPrepareRequestReceived)  3--> Meaning,1.0003=29
I***(ProposalPreparedNotificationSent)  Meaning,1.0003<--(1.0001,42) == SUCCESS
```

3-Prepare-3-Accept

Case 4:1 Learns Twice

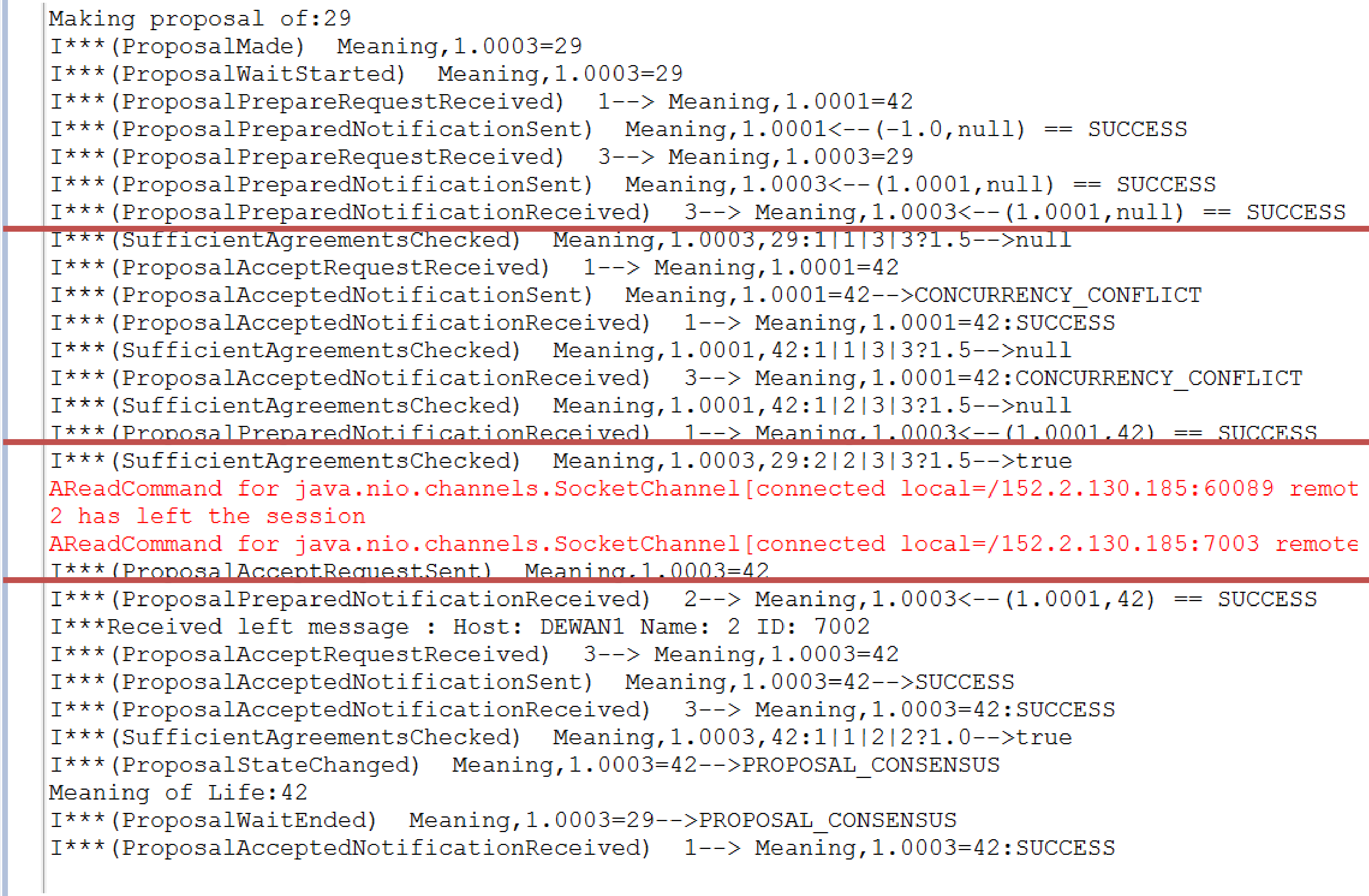
1



```
Making proposal of:42
I***(ProposalMade)  Meaning,1.0001=42
I***(ProposalWaitStarted)  Meaning,1.0001=42
I***(ProposalPrepareRequestReceived)  1--> Meaning,1.0001=42
I***(ProposalPreparedNotificationSent)  Meaning,1.0001<--(-1.0,null) == SUCCESS
I***(ProposalPreparedNotificationReceived)  1--> Meaning,1.0001<--(-1.0,null) == SUCCESS
I***(SufficientAgreementsChecked)  Meaning,1.0001,42:1|1|3|3?1.5-->null
I***(ProposalPreparedNotificationReceived)  2--> Meaning,1.0001<--(-1.0,null) == SUCCESS
I***(SufficientAgreementsChecked)  Meaning,1.0001,42:2|2|3|3?1.5-->true
I***(ProposalAcceptRequestSent)  Meaning,1.0001=42
I***(ProposalPreparedNotificationReceived)  3--> Meaning,1.0001<--(-1.0,null) == SUCCESS
I***(ProposalAcceptRequestReceived)  1--> Meaning,1.0001=42
I***(ProposalAcceptedNotificationSent)  Meaning,1.0001=42-->SUCCESS
I***(ProposalAcceptedNotificationReceived)  1--> Meaning,1.0001=42:SUCCESS
I***(SufficientAgreementsChecked)  Meaning,1.0001,42:1|1|3|3?1.5-->null
I***(ProposalAcceptedNotificationReceived)  3--> 
I***(SufficientAgreementsChecked)  Meaning,1.0001
I***(ProposalAcceptedNotificationReceived)  2--> 
I***(SufficientAgreementsChecked)  Meaning,1.0001
I***(ProposalStateChanged)  Meaning,1.0001=42-->P
Meaning of Life:42
I***(ProposalWaitEnded)  Meaning,1.0001=42-->PROP
I***(ProposalPrepareRequestReceived)  3--> Meaning,1.0003=29
I***(ProposalPreparedNotificationSent)  Meaning,1.0003<--(1.0001,42) == SUCCESS
AReadCommand for java.nio.channels.SocketChannel[connected local=/152.2.130.185:60079 rem
2 has left the session
AReadCommand for java.nio.channels.SocketChannel[connected local=/152.2.130.185:7001 remo
I***Received left message : Host: DEWAN1 Name: 2 ID: 7002
I***(ProposalAcceptRequestReceived)  3--> Meaning,1.0003=42
I***(ProposalAcceptedNotificationSent)  Meaning,1.0003=42-->SUCCESS
I***(ProposalAcceptedNotificationReceived)  3--> Meaning,1.0003=42:SUCCESS
I***(SufficientAgreementsChecked)  Meaning,1.0003,42:1|1|2|2?1.0-->true
I***(ProposalStateChanged)  Meaning,1.0003=42-->PROPOSAL_CONSENSUS
Meaning of Life:42
I***(ProposalAcceptedNotificationReceived)  1--> Meaning,1.0003=42:SUCCESS
```

Two read failures as each process creates a connection to the other

Case 4: 3 Learns Once



```
Making proposal of:29
I***(ProposalMade)  Meaning,1.0003=29
I***(ProposalWaitStarted)  Meaning,1.0003=29
I***(ProposalPrepareRequestReceived)  1--> Meaning,1.0001=42
I***(ProposalPreparedNotificationSent)  Meaning,1.0001<--(-1.0,null) == SUCCESS
I***(ProposalPrepareRequestReceived)  3--> Meaning,1.0003=29
I***(ProposalPreparedNotificationSent)  Meaning,1.0003<--(1.0001,null) == SUCCESS
I***(ProposalPreparedNotificationReceived)  3--> Meaning,1.0003<--(1.0001,null) == SUCCESS
I***(SufficientAgreementsChecked)  Meaning,1.0003,29:1|1|3|3?1.5-->null
I***(ProposalAcceptRequestReceived)  1--> Meaning,1.0001=42
I***(ProposalAcceptedNotificationSent)  Meaning,1.0001=42-->CONCURRENCY_CONFLICT
I***(ProposalAcceptedNotificationReceived)  1--> Meaning,1.0001=42:SUCCESS
I***(SufficientAgreementsChecked)  Meaning,1.0001,42:1|1|3|3?1.5-->null
I***(ProposalAcceptedNotificationReceived)  3--> Meaning,1.0001=42:CONCURRENCY_CONFLICT
I***(SufficientAgreementsChecked)  Meaning,1.0001,42:1|2|3|3?1.5-->null
I***(ProposalPreparedNotificationReceived)  1--> Meaning,1.0003<--(1.0001,42) == SUCCESS
I***(SufficientAgreementsChecked)  Meaning,1.0003,29:2|2|3|3?1.5-->true
AReadCommand for java.nio.channels.SocketChannel[connected local=/152.2.130.185:60089 remot
2 has left the session
AReadCommand for java.nio.channels.SocketChannel[connected local=/152.2.130.185:7003 remote
I***(ProposalAcceptRequestSent)  Meaning,1.0003=42
I***(ProposalPreparedNotificationReceived)  2--> Meaning,1.0003<--(1.0001,42) == SUCCESS
I***Received left message : Host: DEWAN1 Name: 2 ID: 7002
I***(ProposalAcceptRequestReceived)  3--> Meaning,1.0003=42
I***(ProposalAcceptedNotificationSent)  Meaning,1.0003=42-->SUCCESS
I***(ProposalAcceptedNotificationReceived)  3--> Meaning,1.0003=42:SUCCESS
I***(SufficientAgreementsChecked)  Meaning,1.0003,42:1|1|2|2?1.0-->true
I***(ProposalStateChanged)  Meaning,1.0003=42-->PROPOSAL_CONSENSUS
Meaning of Life:42
I***(ProposalWaitEnded)  Meaning,1.0003=29-->PROPOSAL_CONSENSUS
I***(ProposalAcceptedNotificationReceived)  1--> Meaning,1.0003=42:SUCCESS
```

Why Re-propose with Minority

Proposer (re) proposes value of highest accept proposal number as its own value

The value with highest proposal number may or may not be or become majority value

If it does become majority value then prepare phase may have locked some nodes from accepting it

Case 5: Properties

Node 1’s minority proposal will become majority value

3’s prepare phase has locked itself from accepting it

CASE 5: CONSTRUCTION

Break in sendPrepareFrom3() before sending to 1,2, start of startAcceptPhase()

1-Prepare-*

3-Prepare-3

1-Accept-1,2,3

Resume 1.startAcceptPhase()

3's prepare sees acceptance from 1 and rejection from 3 though 2 has accepted to create majority value

3-Prepare-1

Step Over sendPrepareFrom3()

3-Accept-*

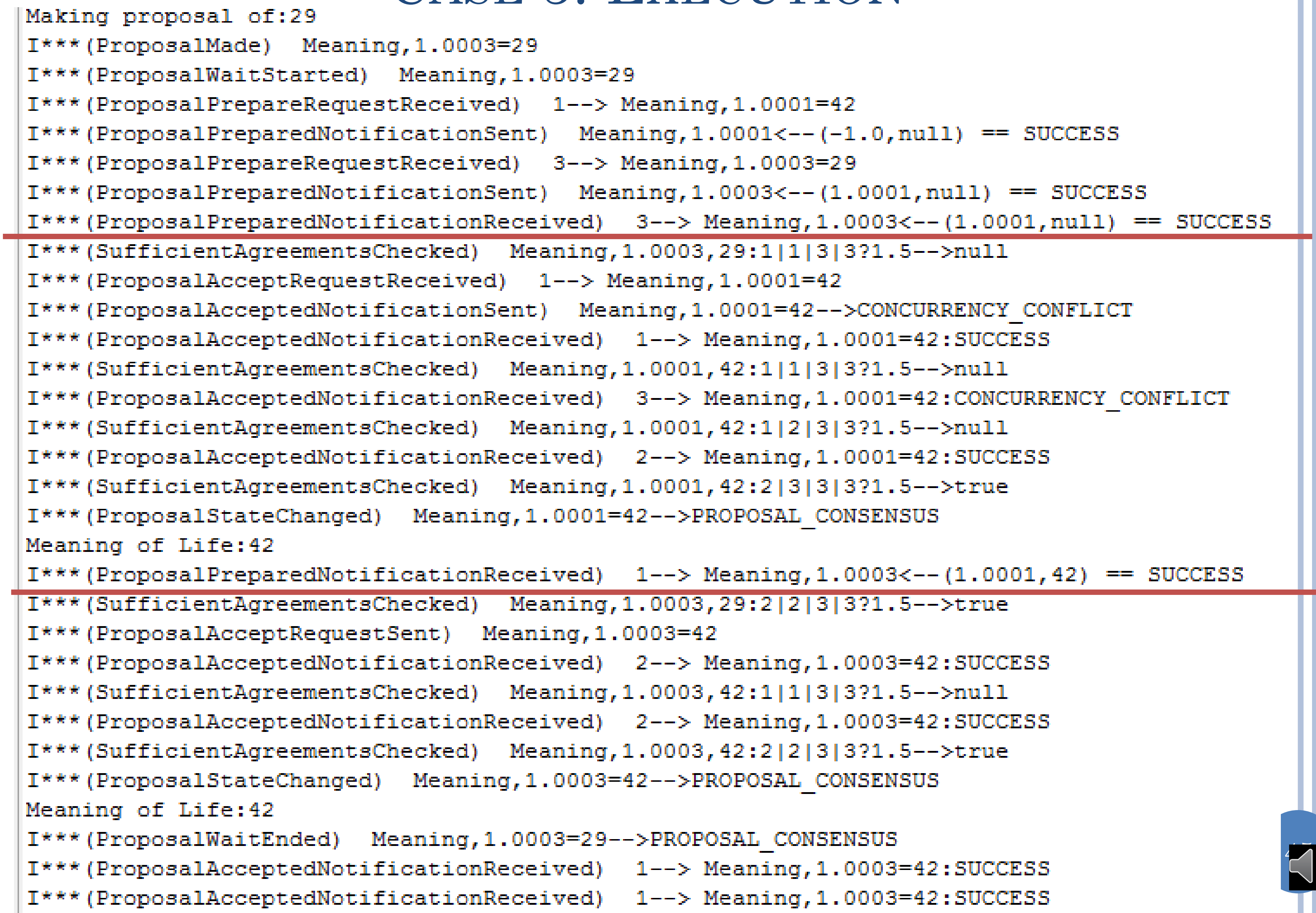
3-Prepare-2

Resume 3.sendAcceptFrom3() and sendPrepareFrom3()

So 3 should re propose 1's value

3

Case 5: Execution



```
Making proposal of:29
I***(ProposalMade)  Meaning,1.0003=29
I***(ProposalWaitStarted)  Meaning,1.0003=29
I***(ProposalPrepareRequestReceived)  1--> Meaning,1.0001=42
I***(ProposalPreparedNotificationSent)  Meaning,1.0001<--(-1.0,null) == SUCCESS
I***(ProposalPrepareRequestReceived)  3--> Meaning,1.0003=29
I***(ProposalPreparedNotificationSent)  Meaning,1.0003<--(1.0001,null) == SUCCESS
I***(ProposalPreparedNotificationReceived)  3--> Meaning,1.0003<--(1.0001,null) == SUCCESS
I***(SufficientAgreementsChecked)  Meaning,1.0003,29:1|1|3|3?1.5-->null
I***(ProposalAcceptRequestReceived)  1--> Meaning,1.0001=42
I***(ProposalAcceptedNotificationSent)  Meaning,1.0001=42-->CONCURRENCY_CONFLICT
I***(ProposalAcceptedNotificationReceived)  1--> Meaning,1.0001=42:SUCCESS
I***(SufficientAgreementsChecked)  Meaning,1.0001,42:1|1|3|3?1.5-->null
I***(ProposalAcceptedNotificationReceived)  3--> Meaning,1.0001=42:CONCURRENCY_CONFLICT
I***(SufficientAgreementsChecked)  Meaning,1.0001,42:1|2|3|3?1.5-->null
I***(ProposalAcceptedNotificationReceived)  2--> Meaning,1.0001=42:SUCCESS
I***(SufficientAgreementsChecked)  Meaning,1.0001,42:2|3|3|3?1.5-->true
I***(ProposalStateChanged)  Meaning,1.0001=42-->PROPOSAL_CONSENSUS
Meaning of Life:42
I***(ProposalPreparedNotificationReceived)  1--> Meaning,1.0003<--(1.0001,42) == SUCCESS
I***(SufficientAgreementsChecked)  Meaning,1.0003,29:2|2|3|3?1.5-->true
I***(ProposalAcceptRequestSent)  Meaning,1.0003=42
I***(ProposalAcceptedNotificationReceived)  2--> Meaning,1.0003=42:SUCCESS
I***(SufficientAgreementsChecked)  Meaning,1.0003,42:1|1|3|3?1.5-->null
I***(ProposalAcceptedNotificationReceived)  2--> Meaning,1.0003=42:SUCCESS
I***(SufficientAgreementsChecked)  Meaning,1.0003,42:2|2|3|3?1.5-->true
I***(ProposalStateChanged)  Meaning,1.0003=42-->PROPOSAL_CONSENSUS
Meaning of Life:42
I***(ProposalWaitEnded)  Meaning,1.0003=29-->PROPOSAL_CONSENSUS
I***(ProposalAcceptedNotificationReceived)  1--> Meaning,1.0003=42:SUCCESS
I***(ProposalAcceptedNotificationReceived)  1--> Meaning,1.0003=42:SUCCESS
```

Case 6: Properties

Node 1's minority proposal will not become majority value

But 3 does not know that at start of accept phase, so re-proposes

Construction

Break in sendPrepareFrom3() before sending to 1,2, and start of sendAcceptFrom1() and sendAcceptFrom3()

1-Prepare-*

3-Prepare-3

1-Accept-1

Step Over 1.sendAcceptFrom1()

3's prepare sees acceptance from 1 and rejection from 3

3-Prepare-*

Resume sendPrepareFrom3()

3 re propose 1's value,

3-Accept-*

Resume 3.sendAcceptFrom3()

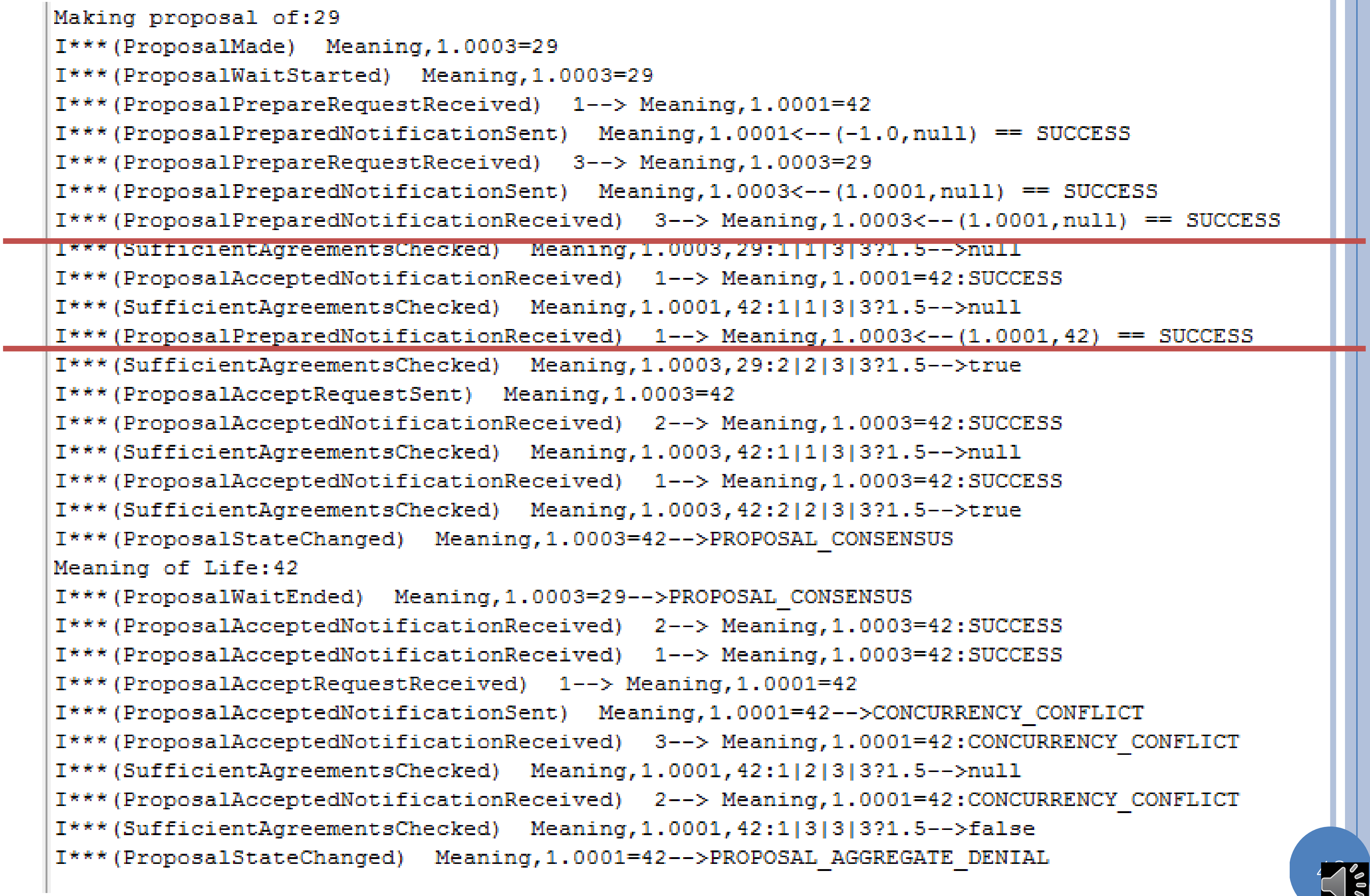
1-Accept-2, 3

Resume sendAcceptFrom1()

Though 2 will reject 1's value as 3 does not know in what order the acceptances will reach 2

3

Case 6: Execution



```
Making proposal of:29
I***(ProposalMade)  Meaning,1.0003=29
I***(ProposalWaitStarted)  Meaning,1.0003=29
I***(ProposalPrepareRequestReceived)  1--> Meaning,1.0001=42
I***(ProposalPreparedNotificationSent)  Meaning,1.0001<--(-1.0,null) == SUCCESS
I***(ProposalPrepareRequestReceived)  3--> Meaning,1.0003=29
I***(ProposalPreparedNotificationSent)  Meaning,1.0003<--(1.0001,null) == SUCCESS
I***(ProposalPreparedNotificationReceived)  3--> Meaning,1.0003<--(1.0001,null) == SUCCESS
I***(SufficientAgreementsChecked)  Meaning,1.0003,29:1|1|3|3?1.5-->null
I***(ProposalAcceptedNotificationReceived)  1--> Meaning,1.0001=42:SUCCESS
I***(SufficientAgreementsChecked)  Meaning,1.0001,42:1|1|3|3?1.5-->null
I***(ProposalPreparedNotificationReceived)  1--> Meaning,1.0003<--(1.0001,42) == SUCCESS
I***(SufficientAgreementsChecked)  Meaning,1.0003,29:2|2|3|3?1.5-->true
I***(ProposalAcceptRequestSent)  Meaning,1.0003=42
I***(ProposalAcceptedNotificationReceived)  2--> Meaning,1.0003=42:SUCCESS
I***(SufficientAgreementsChecked)  Meaning,1.0003,42:1|1|3|3?1.5-->null
I***(ProposalAcceptedNotificationReceived)  1--> Meaning,1.0003=42:SUCCESS
I***(SufficientAgreementsChecked)  Meaning,1.0003,42:2|2|3|3?1.5-->true
I***(ProposalStateChanged)  Meaning,1.0003=42-->PROPOSAL_CONSENSUS
Meaning of Life:42
I***(ProposalWaitEnded)  Meaning,1.0003=29-->PROPOSAL_CONSENSUS
I***(ProposalAcceptedNotificationReceived)  2--> Meaning,1.0003=42:SUCCESS
I***(ProposalAcceptedNotificationReceived)  1--> Meaning,1.0003=42:SUCCESS
I***(ProposalAcceptRequestReceived)  1--> Meaning,1.0001=42
I***(ProposalAcceptedNotificationSent)  Meaning,1.0001=42-->CONCURRENCY_CONFLICT
I***(ProposalAcceptedNotificationReceived)  3--> Meaning,1.0001=42:CONCURRENCY_CONFLICT
I***(SufficientAgreementsChecked)  Meaning,1.0001,42:1|2|3|3?1.5-->null
I***(ProposalAcceptedNotificationReceived)  2--> Meaning,1.0001=42:CONCURRENCY_CONFLICT
I***(SufficientAgreementsChecked)  Meaning,1.0001,42:1|3|3|3?1.5-->false
I***(ProposalStateChanged)  Meaning,1.0001=42-->PROPOSAL_AGGREGATE_DENIAL
```

Paxos Algorithm Property

Re-proposals can prevent any successful acceptance

Case 7 Properties

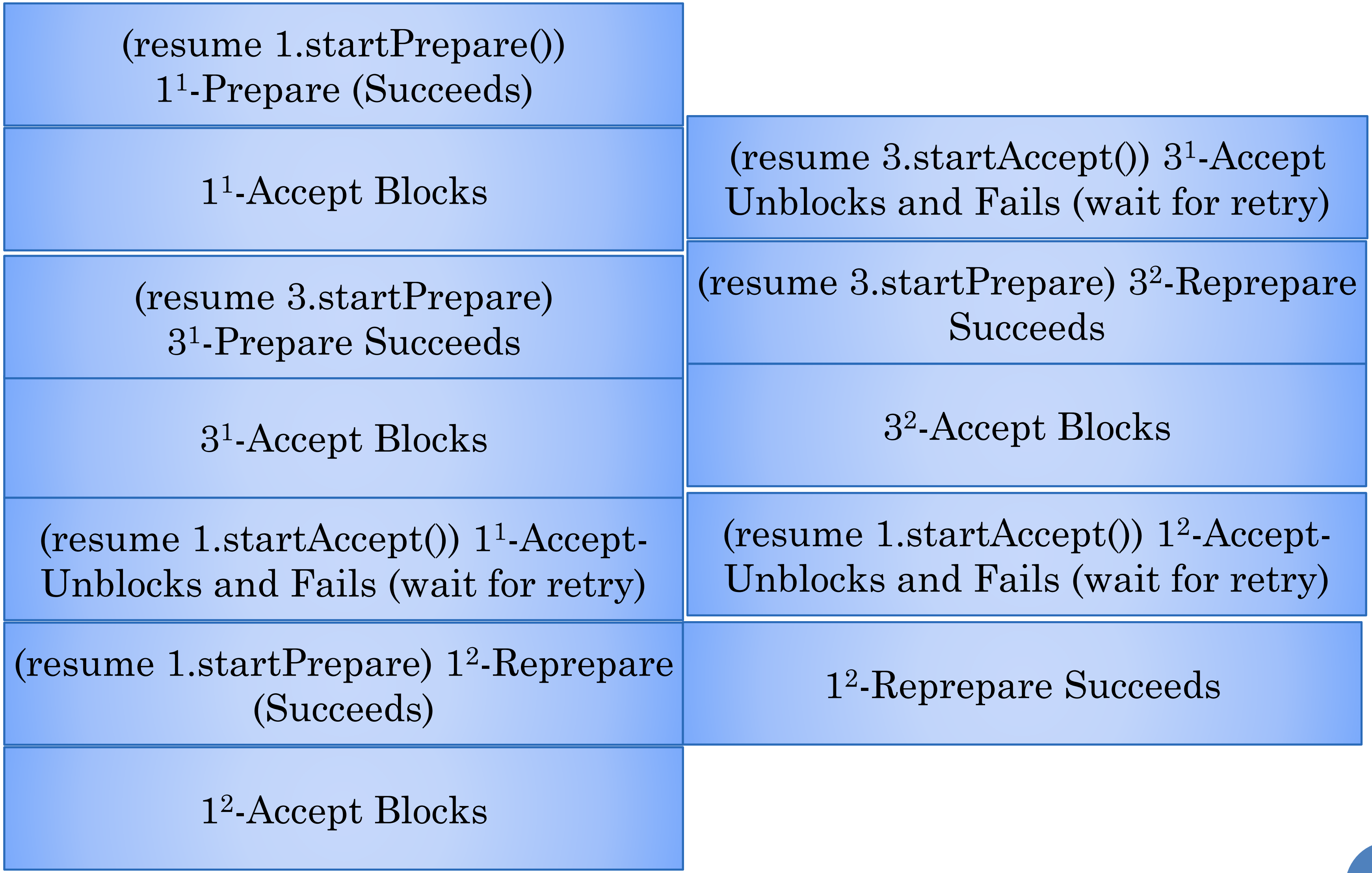
1 ‘s first acceptance prevented from 3’s first prepare

3’s first acceptance is prevented from 1’s second prepare

1’s second acceptance is prevented from 3’s second prepare

3’s secnd acceptance is prevented from 1’s third prepare

Case 7: Paxos Livelock with Retries



(resume 1.startPrepare()) 1^1-Prepare (Succeeds)

1^1-Accept Blocks

(resume 3.startPrepare) 3^1-Prepare Succeeds

3^1-Accept Blocks

(resume 1.startAccept()) 1^1-Accept-Unblocks and Fails (wait for retry)

(resume 1.startPrepare) 1^2-Reprepare (Succeeds)

1^2-Accept Blocks

(resume 3.startAccept()) 3^1-Accept Unblocks and Fails (wait for retry)

(resume 3.startPrepare) 3^2-Reprepare Succeeds

3^2-Accept Blocks

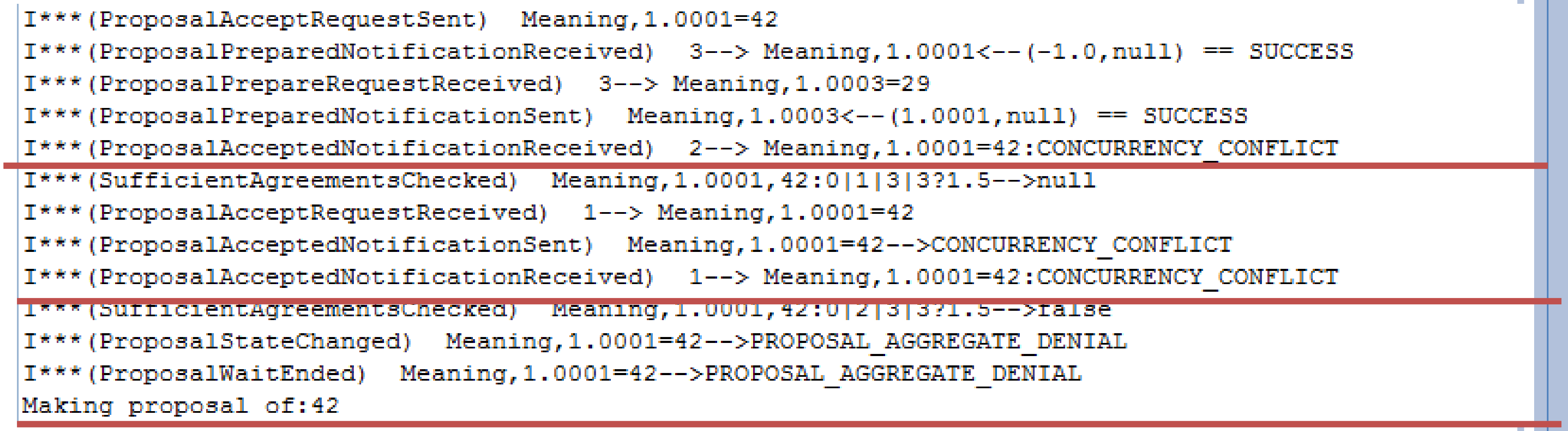
(resume 1.startAccept()) 1^2-Accept-Unblocks and Fails (wait for retry)

1^2-Reprepare Succeeds

Breaks in startPreparePhase() and startAcceptPhase() in 1 and 3

Case 7: 3-Prepare 1-Accept

1



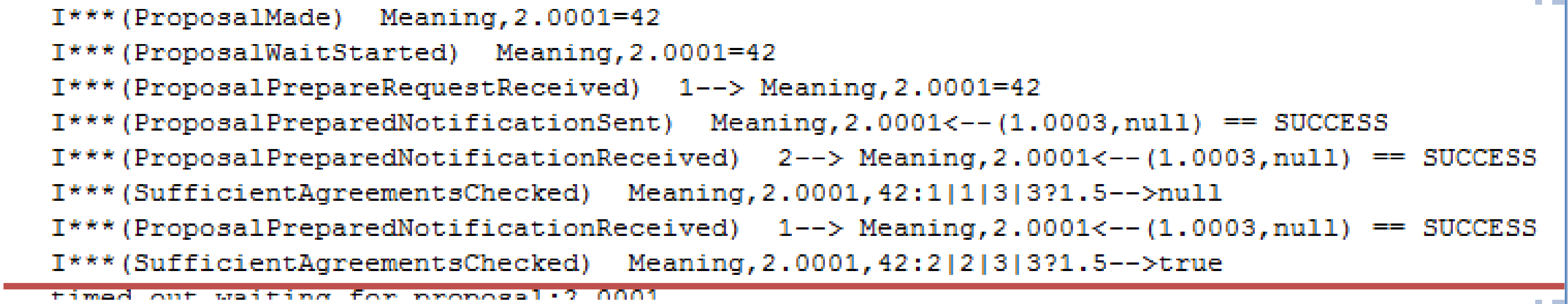
```
I***(ProposalAcceptRequestSent)  Meaning,1.0001=42
I***(ProposalPreparedNotificationReceived)  3--> Meaning,1.0001<--(-1.0,null) == SUCCESS
I***(ProposalPrepareRequestReceived)  3--> Meaning,1.0003=29
I***(ProposalPreparedNotificationSent)  Meaning,1.0003<--(1.0001,null) == SUCCESS
I***(ProposalAcceptedNotificationReceived)  2--> Meaning,1.0001=42:CONCURRENCY_CONFLICT
I***(SufficientAgreementsChecked)  Meaning,1.0001,42:0|1|3|3?1.5-->null
I***(ProposalAcceptRequestReceived)  1--> Meaning,1.0001=42
I***(ProposalAcceptedNotificationSent)  Meaning,1.0001=42-->CONCURRENCY_CONFLICT
I***(ProposalAcceptedNotificationReceived)  1--> Meaning,1.0001=42:CONCURRENCY_CONFLICT
I***(SufficientAgreementsChecked)  Meaning,1.0001,42:0|2|3|3?1.5-->false
I***(ProposalStateChanged)  Meaning,1.0001=42-->PROPOSAL_AGGREGATE_DENIAL
I***(ProposalWaitEnded)  Meaning,1.0001=42-->PROPOSAL_AGGREGATE_DENIAL
Making proposal of:42
```

1-Prepare Succeeds

1-Accept-Unblocks and Fails and 1 retries

Case 7: 1-Reprepare Succeeds

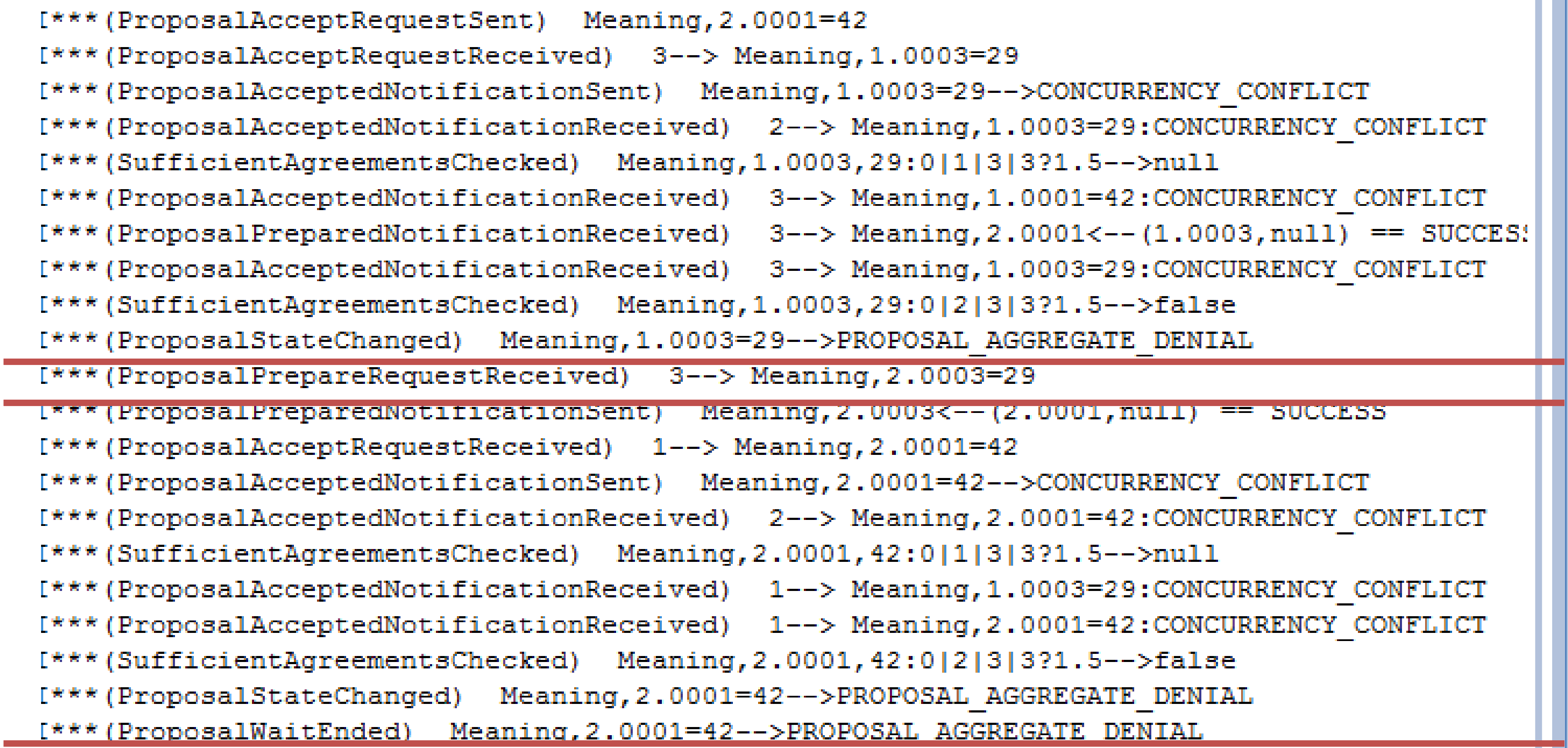
1



```
I***(ProposalMade)  Meaning,2.0001=42
I***(ProposalWaitStarted)  Meaning,2.0001=42
I***(ProposalPrepareRequestReceived)  1--> Meaning,2.0001=42
I***(ProposalPreparedNotificationSent)  Meaning,2.0001<--(1.0003,null) == SUCCESS
I***(ProposalPreparedNotificationReceived)  2--> Meaning,2.0001<--(1.0003,null) == SUCCESS
I***(SufficientAgreementsChecked)  Meaning,2.0001,42:1|1|3|3?1.5-->null
I***(ProposalPreparedNotificationReceived)  1--> Meaning,2.0001<--(1.0003,null) == SUCCESS
I***(SufficientAgreementsChecked)  Meaning,2.0001,42:2|2|3|3?1.5-->true
timed out waiting for proposal:2.0001
```

Case 7: 3-(Re) Accept After 1-(Re-Re) Prepare

1



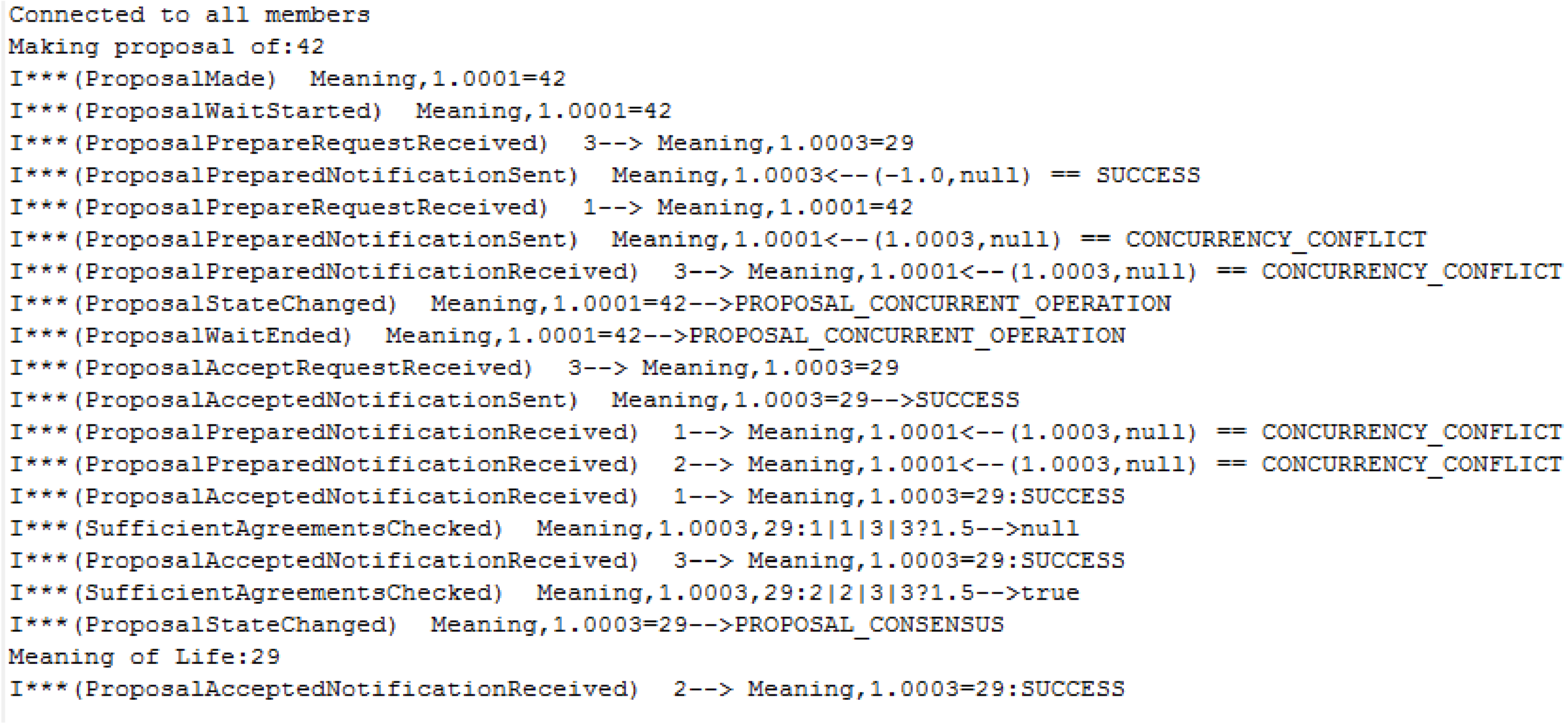
```
[***(ProposalAcceptRequestSent)  Meaning,2.0001=42
[***(ProposalAcceptRequestReceived)  3--> Meaning,1.0003=29
[***(ProposalAcceptedNotificationSent)  Meaning,1.0003=29-->CONCURRENCY_CONFLICT
[***(ProposalAcceptedNotificationReceived)  2--> Meaning,1.0003=29:CONCURRENCY_CONFLICT
[***(SufficientAgreementsChecked)  Meaning,1.0003,29:0|1|3|3?1.5-->null
[***(ProposalAcceptedNotificationReceived)  3--> Meaning,1.0001=42:CONCURRENCY_CONFLICT
[***(ProposalPreparedNotificationReceived)  3--> Meaning,2.0001<--(1.0003,null) == SUCCES
[***(ProposalAcceptedNotificationReceived)  3--> Meaning,1.0003=29:CONCURRENCY_CONFLICT
[***(SufficientAgreementsChecked)  Meaning,1.0003,29:0|2|3|3?1.5-->false
[***(ProposalStateChanged)  Meaning,1.0003=29-->PROPOSAL_AGGREGATE_DENIAL
[***(ProposalPrepareRequestReceived)  3--> Meaning,2.0003=29
[***(ProposalPreparedNotificationSent)  Meaning,2.0003<--(2.0001,null) == SUCCESS
[***(ProposalAcceptRequestReceived)  1--> Meaning,2.0001=42
[***(ProposalAcceptedNotificationSent)  Meaning,2.0001=42-->CONCURRENCY_CONFLICT
[***(ProposalAcceptedNotificationReceived)  2--> Meaning,2.0001=42:CONCURRENCY_CONFLICT
[***(SufficientAgreementsChecked)  Meaning,2.0001,42:0|1|3|3?1.5-->null
[***(ProposalAcceptedNotificationReceived)  1--> Meaning,1.0003=29:CONCURRENCY_CONFLICT
[***(ProposalAcceptedNotificationReceived)  1--> Meaning,2.0001=42:CONCURRENCY_CONFLICT
[***(SufficientAgreementsChecked)  Meaning,2.0001,42:0|2|3|3?1.5-->false
[***(ProposalStateChanged)  Meaning,2.0001=42-->PROPOSAL_AGGREGATE_DENIAL
[***(ProposalWaitEnded)  Meaning,2.0001=42-->PROPOSAL_AGGREGATE_DENIAL
```

3-Accept Fails

3-Reproposes

1-Reaccept Fails

Case 8: Uncontrolled Execution-1 (Conflict in Prepare Phase)

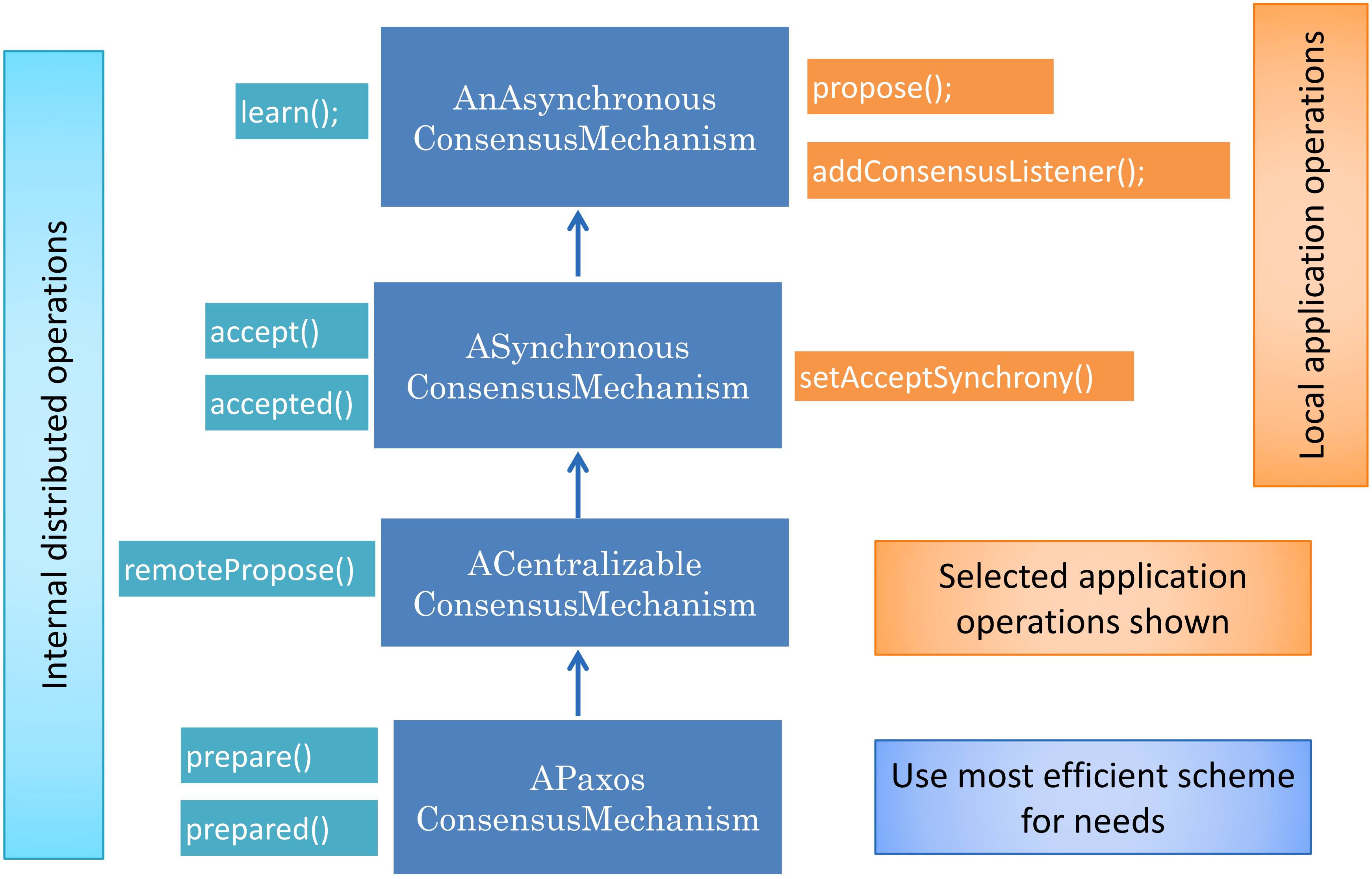


```
Connected to all members
Making proposal of:42
I***(ProposalMade)  Meaning,1.0001=42
I***(ProposalWaitStarted)  Meaning,1.0001=42
I***(ProposalPrepareRequestReceived)  3--> Meaning,1.0003=29
I***(ProposalPreparedNotificationSent)  Meaning,1.0003<--(-1.0,null) == SUCCESS
I***(ProposalPrepareRequestReceived)  1--> Meaning,1.0001=42
I***(ProposalPreparedNotificationSent)  Meaning,1.0001<--(1.0003,null) == CONCURRENCY_CONFLICT
I***(ProposalPreparedNotificationReceived)  3--> Meaning,1.0001<--(1.0003,null) == CONCURRENCY_CONFLICT
I***(ProposalStateChanged)  Meaning,1.0001=42-->PROPOSAL_CONCURRENT_OPERATION
I***(ProposalWaitEnded)  Meaning,1.0001=42-->PROPOSAL_CONCURRENT_OPERATION
I***(ProposalAcceptRequestReceived)  3--> Meaning,1.0003=29
I***(ProposalAcceptedNotificationSent)  Meaning,1.0003=29-->SUCCESS
I***(ProposalPreparedNotificationReceived)  1--> Meaning,1.0001<--(1.0003,null) == CONCURRENCY_CONFLICT
I***(ProposalPreparedNotificationReceived)  2--> Meaning,1.0001<--(1.0003,null) == CONCURRENCY_CONFLICT
I***(ProposalAcceptedNotificationReceived)  1--> Meaning,1.0003=29:SUCCESS
I***(SufficientAgreementsChecked)  Meaning,1.0003,29:1|1|3|3?1.5-->null
I***(ProposalAcceptedNotificationReceived)  3--> Meaning,1.0003=29:SUCCESS
I***(SufficientAgreementsChecked)  Meaning,1.0003,29:2|2|3|3?1.5-->true
I***(ProposalStateChanged)  Meaning,1.0003=29-->PROPOSAL_CONSENSUS
Meaning of Life:29
I***(ProposalAcceptedNotificationReceived)  2--> Meaning,1.0003=29:SUCCESS
```

Paxos vs. Centralized Synchronous

- Multiple client UIs commit to single server
  - Browser-Sakai
- Nested transaction involving multiple logical servers
  - Travelocity (non replicated)
- Physical replication with multiple changers
  - Diff-based with divergence (Git)
  - Snapshot-based (Google Drive, OneDrive)
  - Command-based: replicated state machines (Google Docs, LiveMeeting)
- Lock and other meta/configuration state ✓
  - Live Meeting
- Physical mirroring
  - Akamai
- Master (primary)-slave(backup) replication
- Master-master replication
  - Disjoint writes
  - Overlapping writes

Consensus Mechanism Hierarchy



Internal distributed operations

learn();

AnAsynchronous ConsensusMechanism

propose();

addConsensusListener();

accept()

accepted()

ASynchronous ConsensusMechanism

setAcceptSynchrony()

remotePropose()

ACentralizable ConsensusMechanism

prepare()

prepared()

APaxos ConsensusMechanism

Local application operations

Selected application operations shown

Use most efficient scheme for needs

Customization

```
public interface ConsensusCustomization {
public ConcurrencyKind getConcurrencyKind();
public void setConcurrencyKind(ConcurrencyKind consistencyStrength) ;
public ProposalFeedbackKind getProposalVetoKind();
public void setProposalVetoKind(ProposalFeedbackKind
proposalRejectionKind);
public ReplicationSynchrony getAcceptSynchrony();
public void setAcceptSynchrony(ReplicationSynchrony consensusSynchrony);
public void setSendRejectionInformation(boolean newVal);
public boolean isSendRejectionNotification();
public boolean isAllSynchronous();
public void setAllowVeto(boolean newVal);

public ConsensusMemberSetKind getConsensusMemberSetKind() ;
public void setConsensusMemberSetKind(ConsensusMemberSetKind
consensusMemberSet) ;
public boolean isValueSynchrony();
public void setValueSynchrony(boolean newVal) ;
public boolean isSendAcceptReplyForResolvedProposal();
public void setSendAcceptReplyForResolvedProposal(
boolean newVal) ;
public boolean isClient() ;
public boolean isServer();
public boolean isCentralizedPropose() ;
```

Summary

- Distributed → Replicated Systems
- RPC → Replicated Object
- Pure replication synchrony → Global clock, time bounds
- Replication Synchrony →Propose, Invalid State, Listeners, rather than pure get and set
- Replication Synchrony → Two phase algorithm
- Shared abstraction and algorithms for replication synchrony and distributed proposal rejection
- Degree of replication synchrony and vetoing based on set synchronized with/consulted by proposer
- Safety vs Progress
- Concurrent proposes in asynchronous and synchronous case can be handled with centralized coordinator
  - Centralized Synchronous is Two Phase Commit if invalidation step involves transaction set up and checks
- Coordinator switch-offs can be done using consensus protocol or zero phase id choice
- Central solution can lead to inconsistency even if majority alive
- 3-Phase Paxos supports consistency if majority alive
  - Centralization and timeouts for practical reasons
  - Atomic setting of proposal number and getting of state in acceptor
  - Convergence towards proposals with higher proposal numbers and re-proposing of acceptances
  - To update a value multiple times, multiple paxos mechanisms instantiated, once for each update
  - Centralization to achieve consensus about update sequence number