Formalization of Business Process of Internal Controls

Yasuhito Arimoto

Background

- Definition of Internal Controls:
  A process effected by an organization's structure, work and authority flows, people and management information systems, designed to help the organization accomplish specific goals or objectives.
  - by the Committee of Sponsoring Organizations of Treadway commission (COSO)
- Aims of Internal Controls:
  - Efficacy and efficiency of the business
  - Trustworthiness of the financial reports
  - Compliance with applicable laws and regulations

Establishment of Law

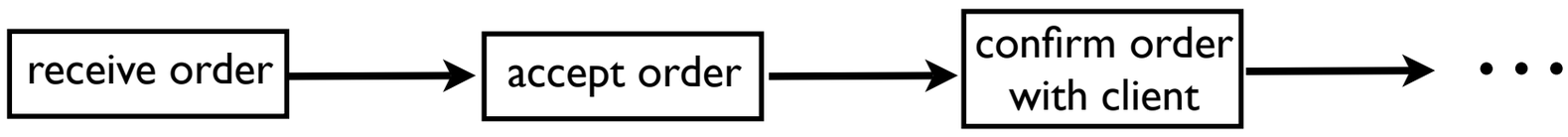
- Sarbanes-Oxley Act (2002, the US) and Financial Instruments Exchange Law (2006, Japan)
  - Evaluation of internal controls by management
  - Audit of the evaluation's report by external auditors
- Evaluation
  - Understanding risks and controls in business processes
  - Checking effectiveness of controls

Background

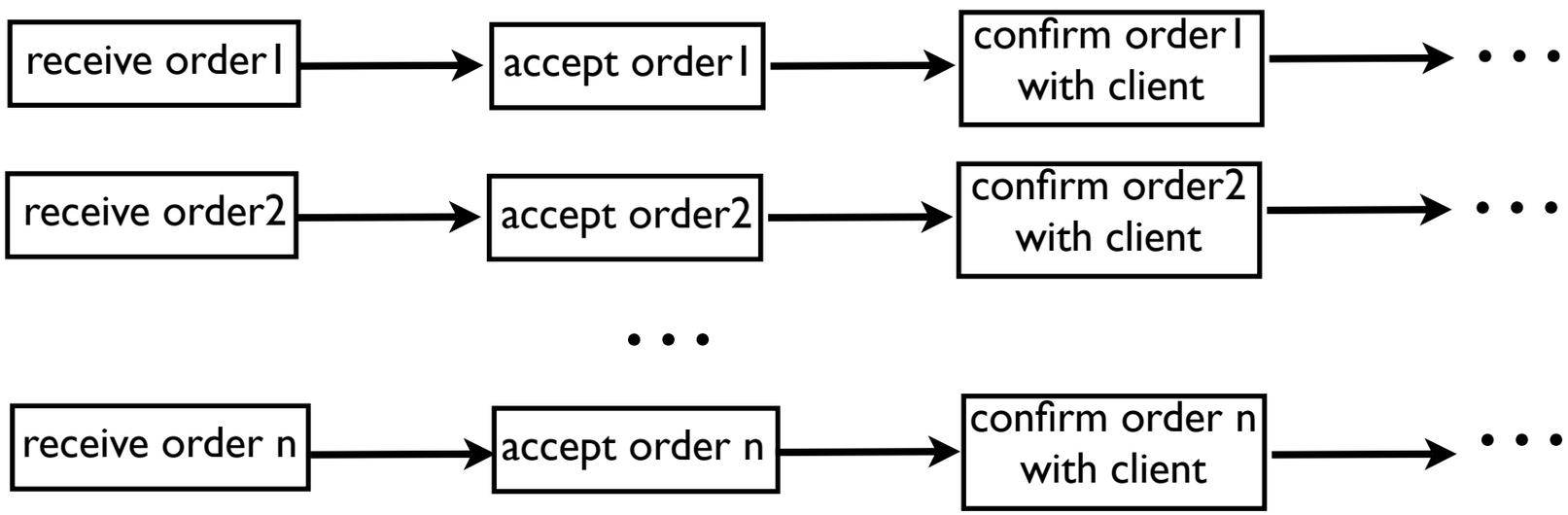
- Sarbanes-Oxley Act (2002, the US) and Financial Instruments Exchange Law (2006, Japan)
  - Evaluation of internal controls by management
  - Audit of the evaluation's report by external auditors
- Evaluation of Internal Controls
  - Understanding risks and controls in business processes
  - Checking effectiveness of controls
  - Business workflow diagrams, risk control matrices, business process narratives are used for the evaluation.
    - A business workflow diagram represents sequences of activities needed in order to fulfill the business goal.
    - A risk control matrix shows relations of risks and controls
    - A business workflow shows detailed information of a business workflow

Background

- An instance of sequence represented by a workflow diagram is called a session
- When a workflow diagram is analyzed, each session is analyzed independently.

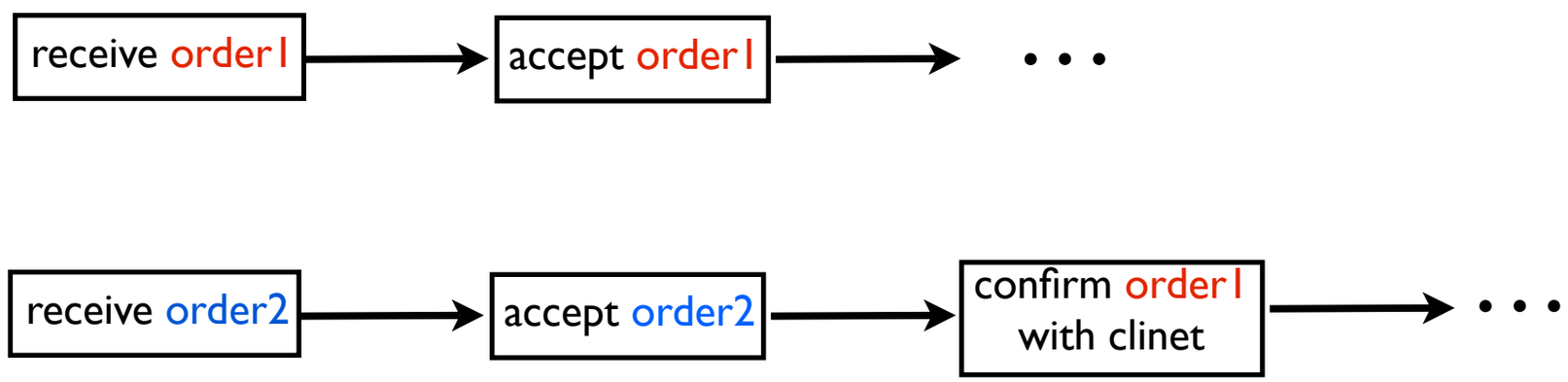


- However, in the real world, each session is executed in parallel with other sessions.
  - For example, while handling orders, employees handle many orders not one after another but in parallel.



Problem

- Risks appear on not only the domain whose range is closed to single sessions, but the one who includes multiple sessions.
  - in single sessions: forging a document in a business process.
  - in multiple sessions: handling a document which is supposed to belong to a wrong session.

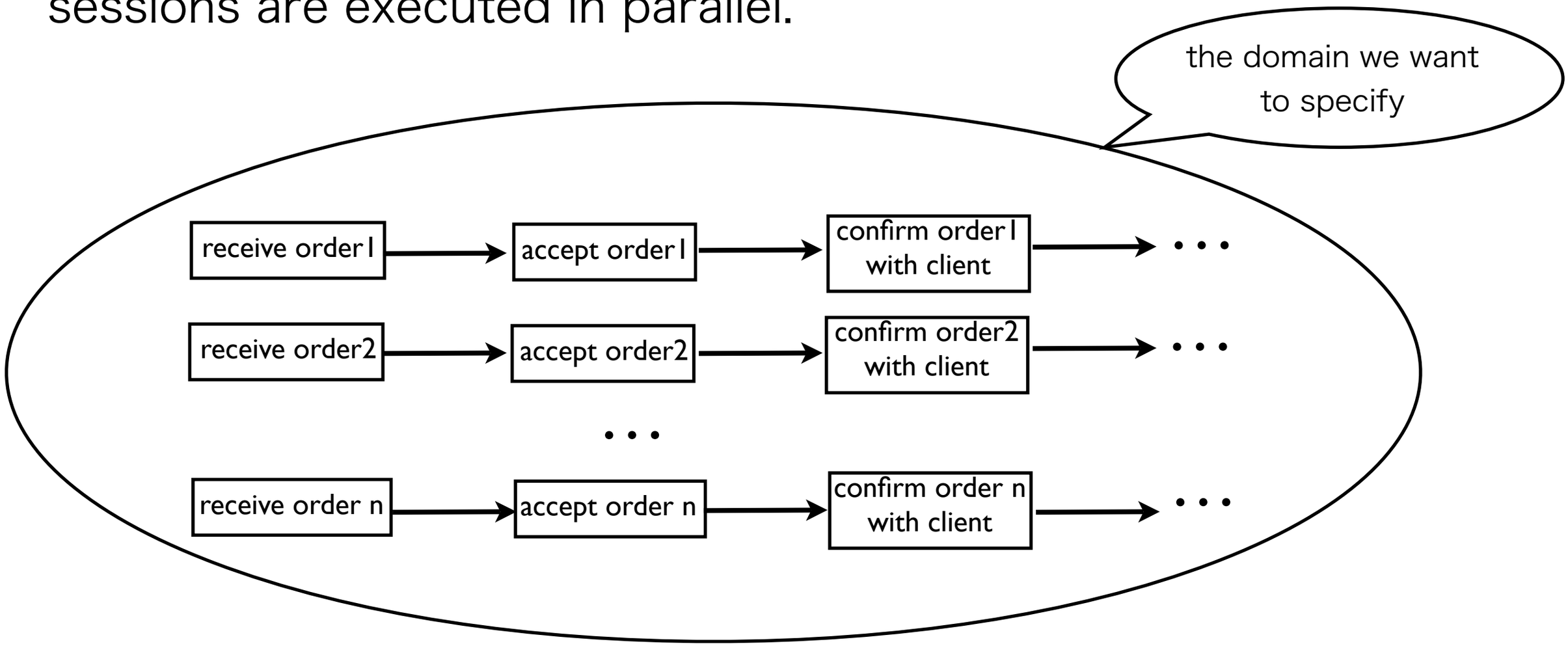


Aim

- Formalization of verification of business processes
  - Formalizing business processes with risks and controls
  - Verification of effectivity of controls
- By formalization of risks and controls, we can classify
  - what are risks and controls in single sessions
  - what are risks and controls in multiple sessions
- By formal verification of business processes,
  - we can check if the specification is described right,
  - we can analyze effectivity of controls on computers

Definition of business process

- A business process is behaviour of a domain in which many sessions are executed in parallel.



Approach

- Formalization of business processes
  - Domain modeling
    - defining entities, events, and behaviours in an domain for a business
    - Entities : departments, documents, etc
    - Events : creating a document, sending a document, etc
    - Behaviours : sequences of events

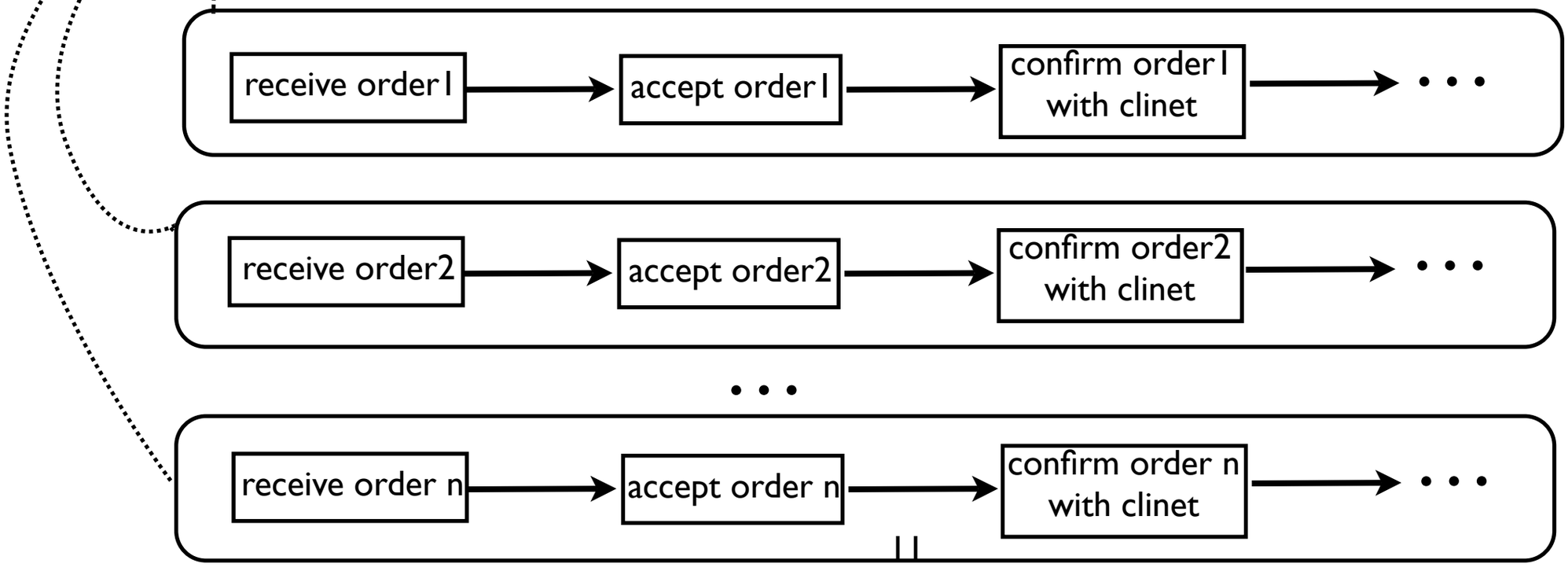
Approach

- Modeling in Document Logic [DL] is applied
- Business activities are recorded in documents.
  - order form, request document, confirmation document, etc
- Focusing on flows of documents
  - Modeling business processes by following flows of documents.
  - Analyzing behaviours which relate to documents.
- Modeling a business processes as a state transition machine, OTS (Observational Transition System)

[DL] S. Iida, G. Denker, and C. Talcott.
Document Logic: Risk Analysis of Business Processes Through Document Authenticity.
In proceedings of 2nd International Workshop on Dynamic and Declarative Business Processes, IEEE, 2009

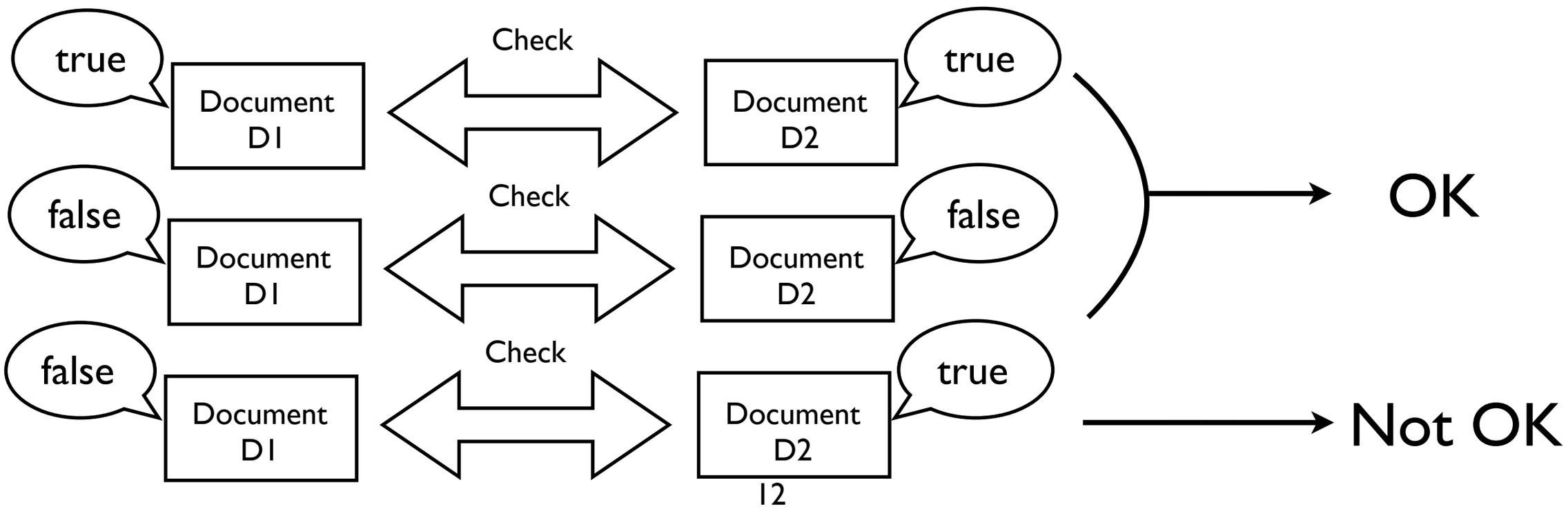
Basic Model of Document

- Attributes of documents
  - Document IDドキュメントのID(Document ID)
  - Document Typeドキュメントの型(DocumentType)
  - Evidence History
  - Division (information to show where the document is)
  - Session ID (Information to show which session the document belongs to)



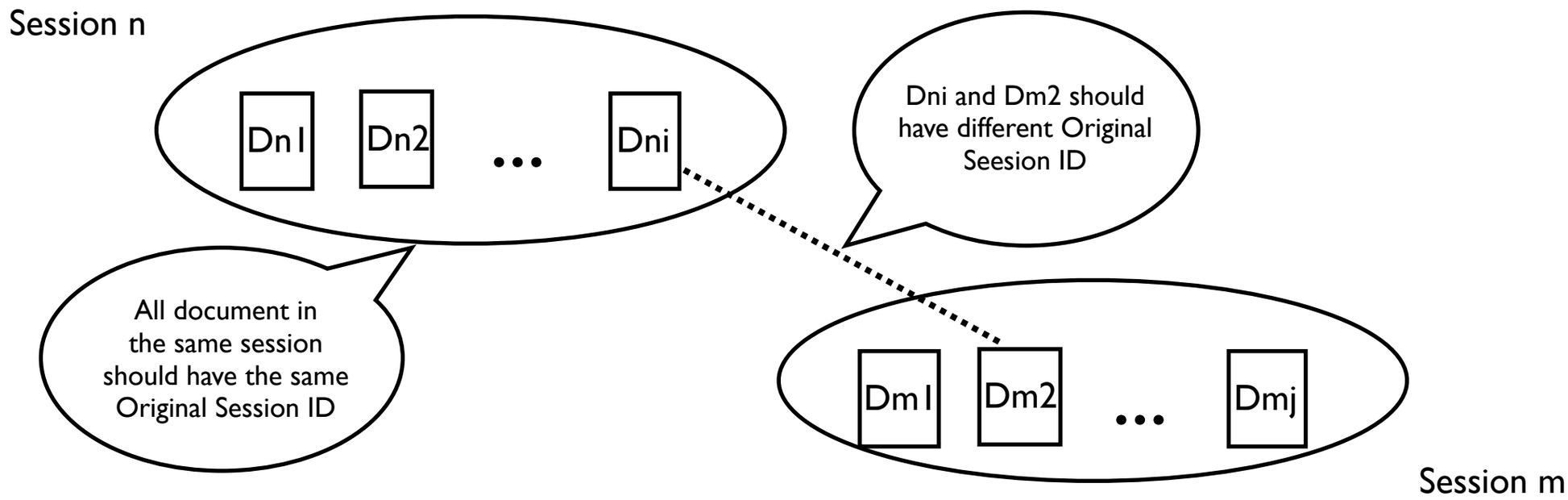
Meta Information for Document

- **Boolean value which represents authenticity of the document**
  - It returns true when the document is not forged
  - It returns false when the document is forged
  - It is referred when a document is checked with another one
  - The one who checks document cannot see the value, but he/she can judge if two values are the same or not



Meta Information for Document

- **Original Session ID**
  - ID of the session where the document is created



Observations

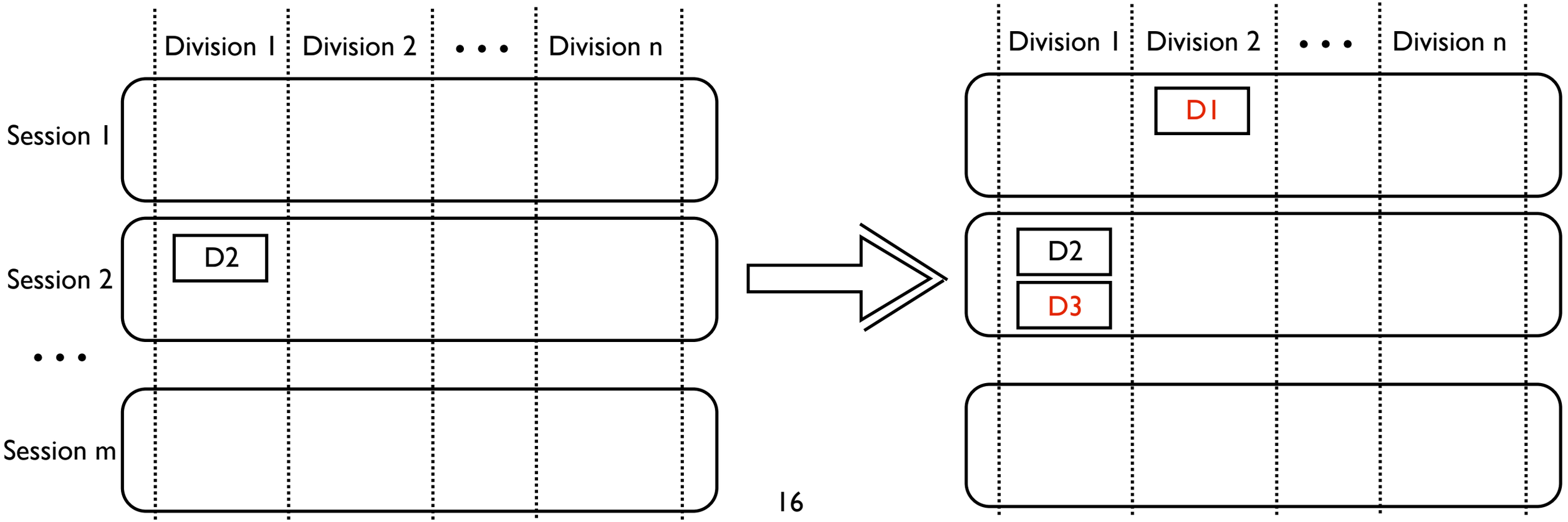
- Basic observations for documents
  - DocumentType : State DocumentID -> DocumentType
  - Evidences : State DocumentID -> EvidenceHistory
  - Place : State DocumentID -> Division
  - SessionID : State DocumentID -> SessionID
- meta information
  - Legal? : State DocumentID -> Bool
  - OriginalID : State DocumentID -> SessionID
- observation for created document IDs
  - DocumentIDList :
    State SessionID DocumentType -> DocumentIDList

Events

- Events can be classified as 3 groups.
  - Regular events
    - Creating a document
    - Sending a document
  - Irregular events
    - Events which make states move to undesirable states.
  - Control events
    - Events which avoid states moving to undesirable states, or which make undesirable states move to desirable states
- Some irregular events and control events only appears on either single sessions or multiple sessions

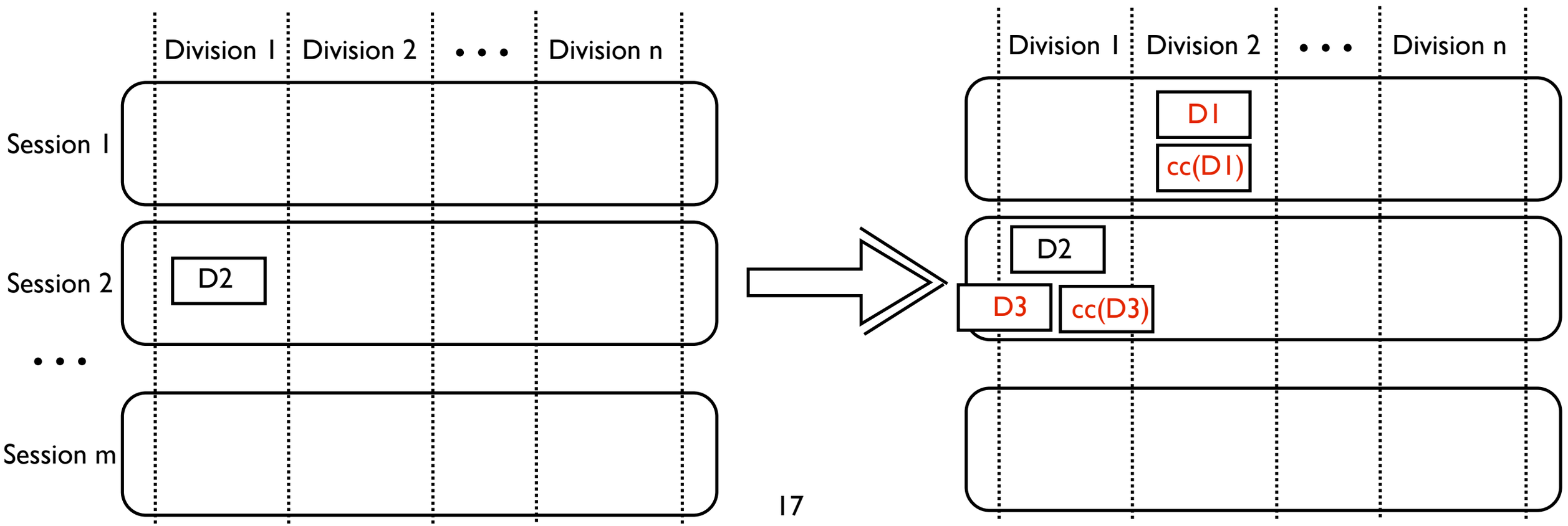
Regular Events

- Creating documents
  - Create-1 : State × SessionID × Division × DocumentID × DocumentType → State
  - Create-2 : State × Division × DocumentID × DocumentType × DocumentID → State
  - The latter one is creating a document from another document.



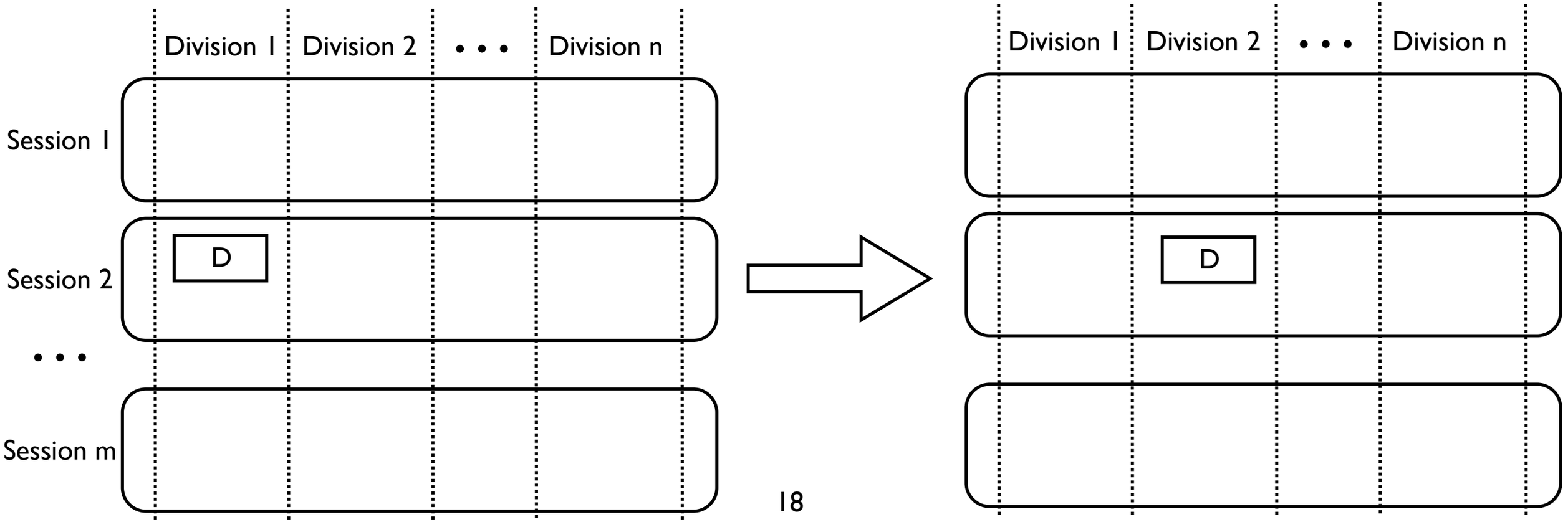
Regular Events

- Creating a document with a carbon copy
  - Create-cc-1 : State × SessionID × Division × DocumentID × DocumentType → State
  - Create-cc-2 : State × Division × DocumentID × DocumentType × DocumentID → State
  - The latter one is creating a document from another document with carbon copy.



Regular Events

- Sending a document
  - Send : State × Division × DocumentID × Division → State
  - A document is sent from a division to another division



Transition Rules

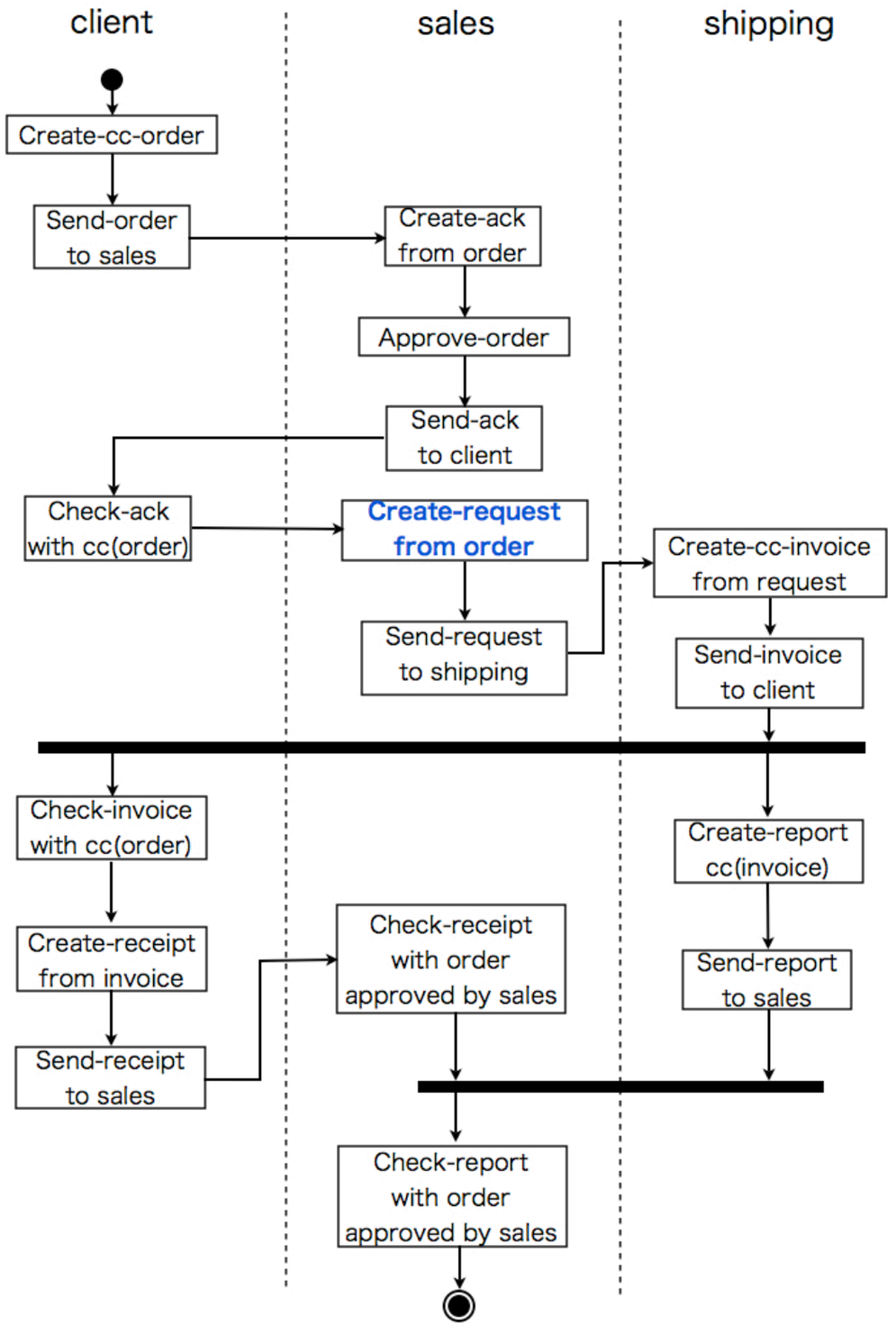
Transition rule for Create-2

```
eq DocumentType(Create-2(S, V, D1, T, D2), D3)
    = if (D1 = D3) then T
      else DocumentType(S, D3) fi .
eq Evidences(Create-2(S, V, D1, T, D2), D3)
    = if (D1 = D3) then emptyE
      else Evidences(S, D3) fi .
eq Place(Create-2(S, V, D1, T, D2), D3)
    = if (D1 = D3) then V
      else Place(S, D3) fi .
eq SessionID(Create-2(S, V, D1, T, D2), D3)
    = if (D1 = D3) then SessionID(S, D2)
      else SessionID(S, D3) fi .
```

```
eq Legal?(Create-2(S, V, D1, T, D2), D3)
    = if (D1 = D3) then Legal?(S, D2)
      else Legal?(S, D3) fi .
eq OriginalID(Create-2(S, V, D1, T, D2), D3)
    = if ((D1 = D3) or (cc(D1) = D3)) then
        SessionID(S, D2)
      else OriginalID(S, D3) fi .
eq DocumentIDList(Create-2(S, V, D1, T1, D2), I, T2)
    = if ((I = SessionID(S, D2)) and (T1 = T2)) then
        (D1 ; DocumentIDList(S, I1, T1))
      else DocumentIDList(S, I, T2) fi .
```

Example: Sales and Ship Process

- Documents
  order, cc(order), ack, request, invoice, cc(invoice), receipt, report
- Example of creating a document
  - Creating request from order
    Create-request : State × DocumentID × DocumentID
    → State
- transition rule
  Create-request(S, D1, D2)
  = Create-2(S, sales, D1, request, D2)
  if c-Create-request(S, D1, D2)
  Create-request(S, D1, D2) = S
  if not(c-Create-request(S, D1, D2))
- c-Create-request(S, D1, D2) is defined as follows
  - D1 is not used as any document ID
  - D2 is in sales
  - Type of D2 is order
  - any documents whose type is request is not created in the session of D2
  - ack is checked with cc(order) in the session of D2



Example: Sales and Ship Process

- Example of sending a document
  - Sending order from client to sales

    Send-order : State × DocumentID → State
    - transition rule

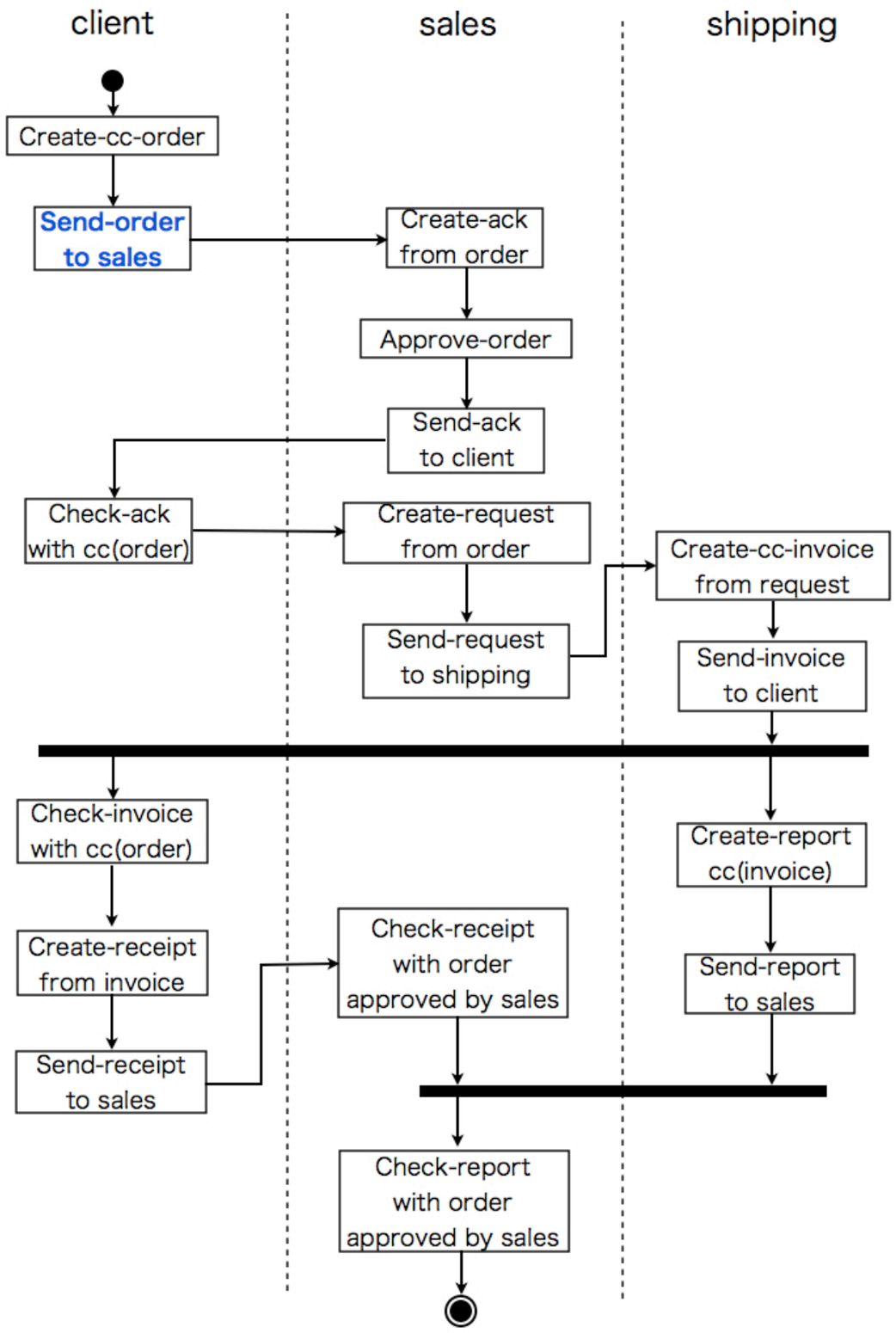
      Send-order(S, D)

      = Send(S, client, D, sales)

      if c-Send-order(S, D)

    Send-order(S, D) = S

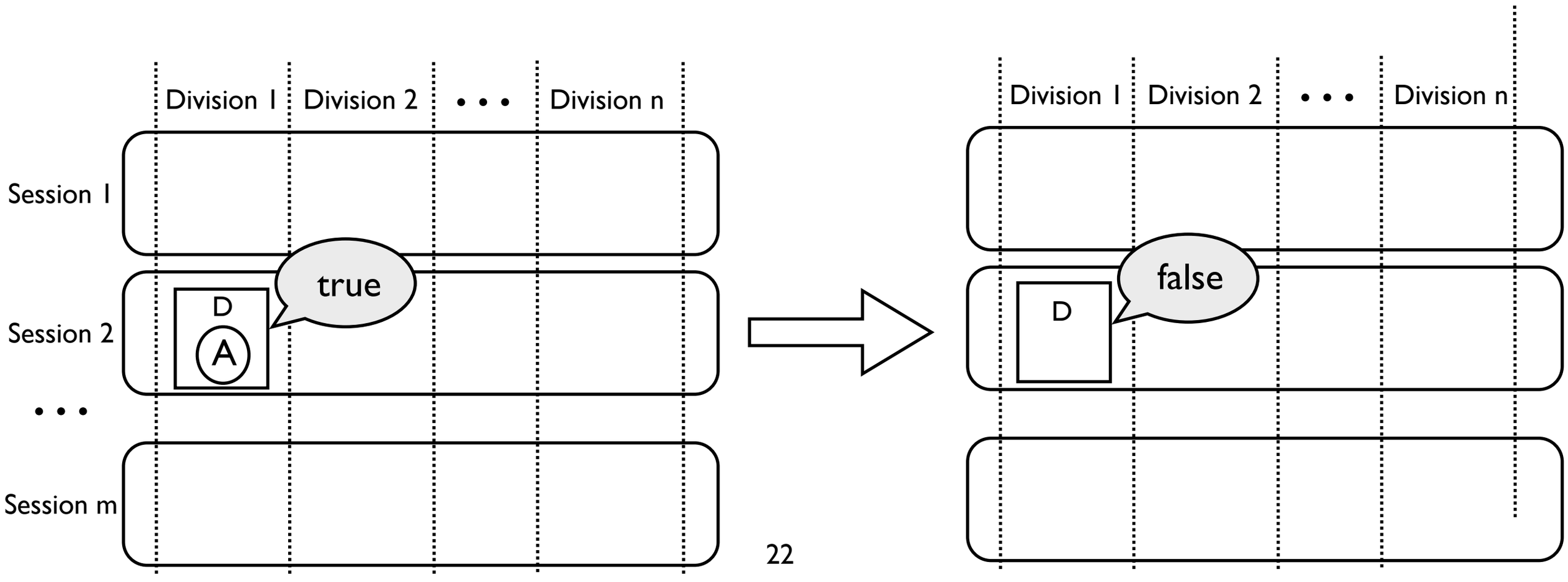
    if not(c-Send-order(S, D))
- c-Send(S, client, D, sales) is as follows
  - D is in client



Irregular Event on Single Session

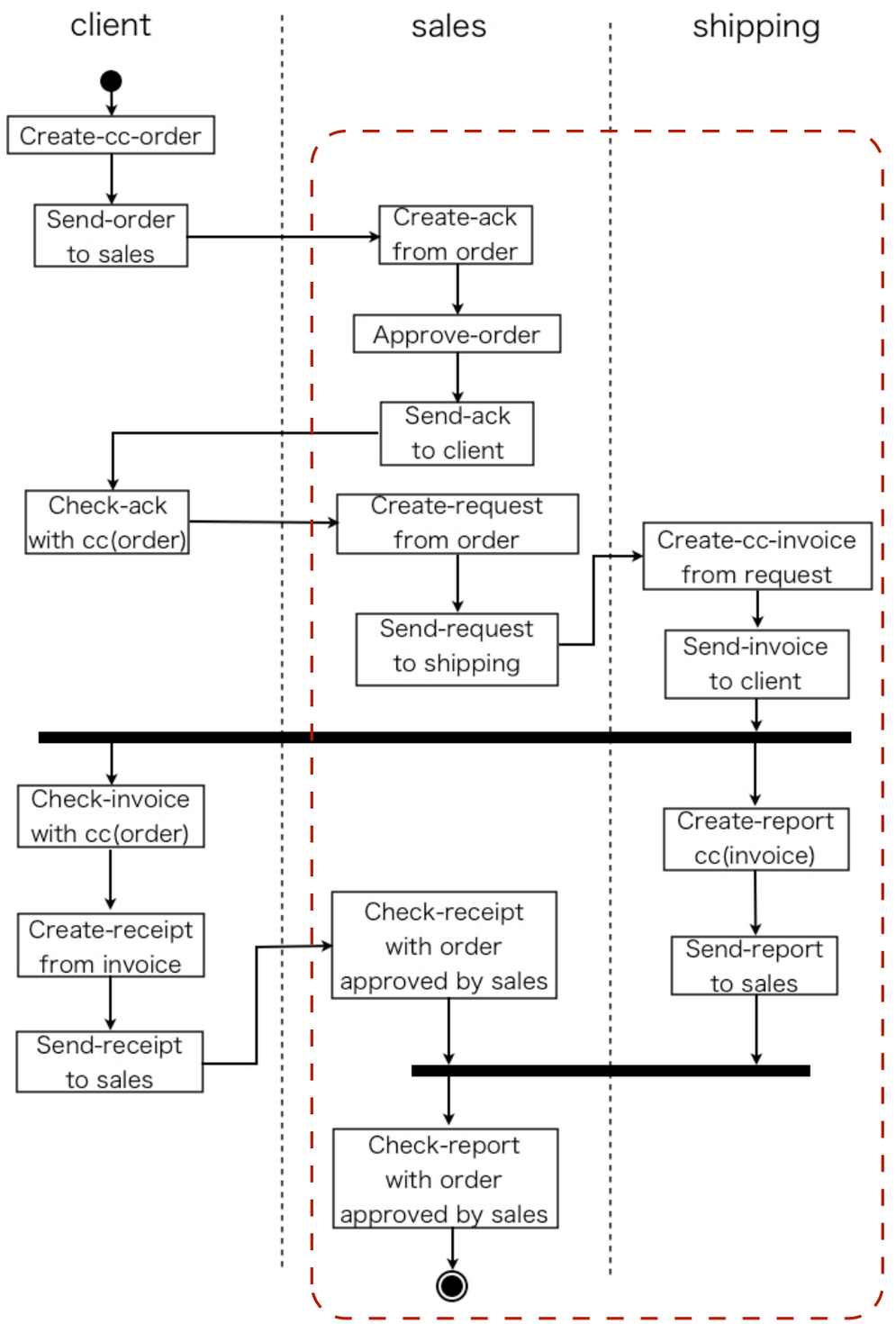
- Forging a document

  Forge : State × DocumentID → State
  - Conditions for forging a document D
    - D is in a untrusted division
    - D is not a carbon copy
    - D is not forged
    - Evidence cannot be forged



Example: Sales and Ship Process

- Forging a document
  - Forge-SaleAndShip : State × DocumentID → State
    - transition rule
      Forge-SaleAndShip(S, D) = Forge(S, client, D, sales)
      if c-Forge-SaleAndShip(S, D)
      Forge-SaleAndShip(S, D) = S
      if not(c-Forge-SaleAnhShip(S, D))
    - c-Forge-SaleAndShip(S, D) is defined as follows
      - D is in an untrusted devision
- We add a new observation for a set of untrusted divisions.
  - UntrustedSet : State → DivisionSet
  - for this process, we suppose sales and shipping are untrusted divisions

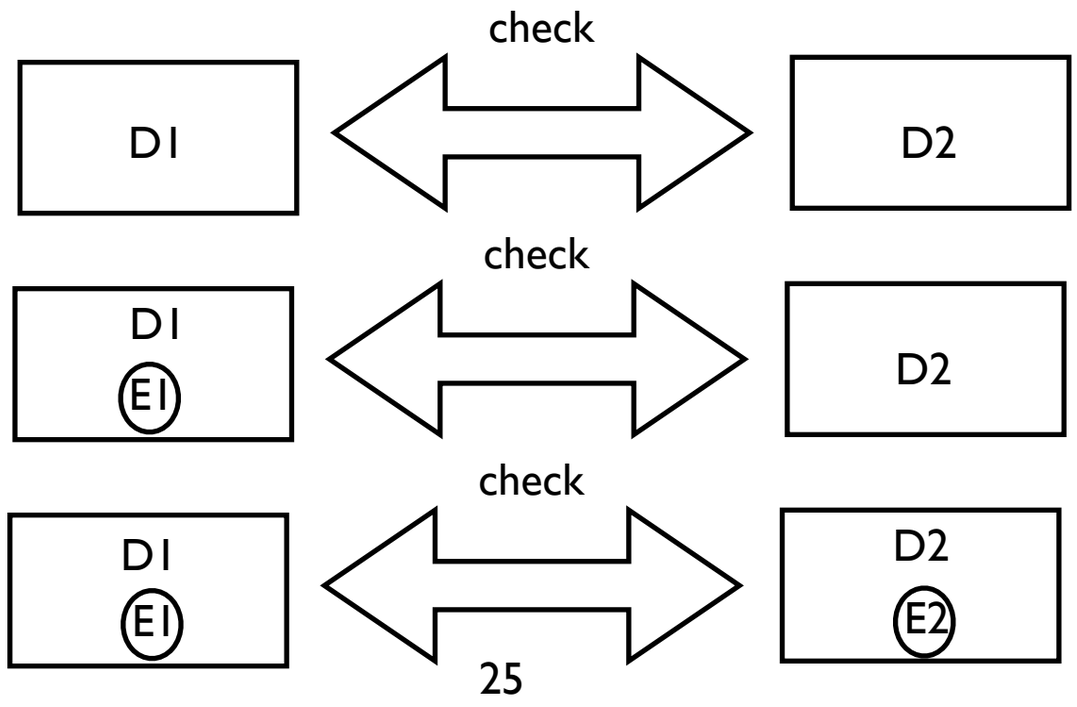


Control Events on Single Session

- Approving a document (putting signature or hanko on a document)
  - Approve-1 : State × Division × DocumentID → State
  - Approve-2 : State × DIvision × DocumentID × Evidence → State
  - The latter one means that approving a document who has an evidence specified by 4th argument.

Control Events on Single Session

- Checking a document with another one
  - Check-1 : Business × Division × DocumentType × DocumentType → Business
  - Check-2 : Business × Division × DocumentType × Evidence × DocumentType → Business
  - Check-3 : Business × Division × DocumentType × DocumentType × Evidence → Business
  - Check-4 : Business × Division × DocumentType × Evidence × DocumentType × Evidence → Business



Example: Sales and Ship Process

- Example of approving a document
  - Approving order

    Approve-order : State × DocumentID → State
    - transition rule

      Approve-order(S, D)

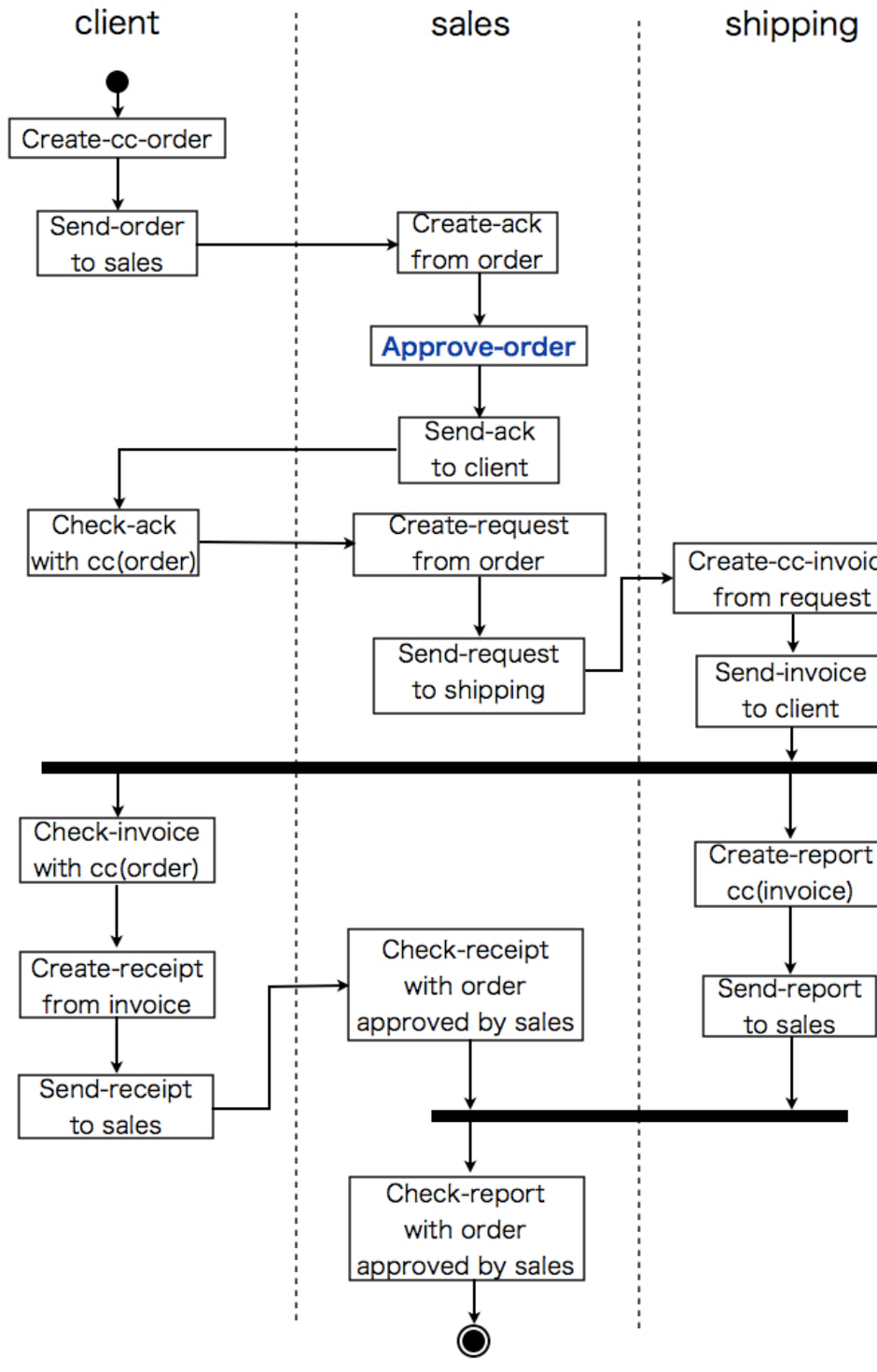
      = Approve-1(S, sales, D)

        if c-Approve-order(S, D)

      Approve-order(S, D) = S

      if not(c-Approve-order(S, D))

- c-Approve-order(S, D) is defined as follows
  - Type of D is order



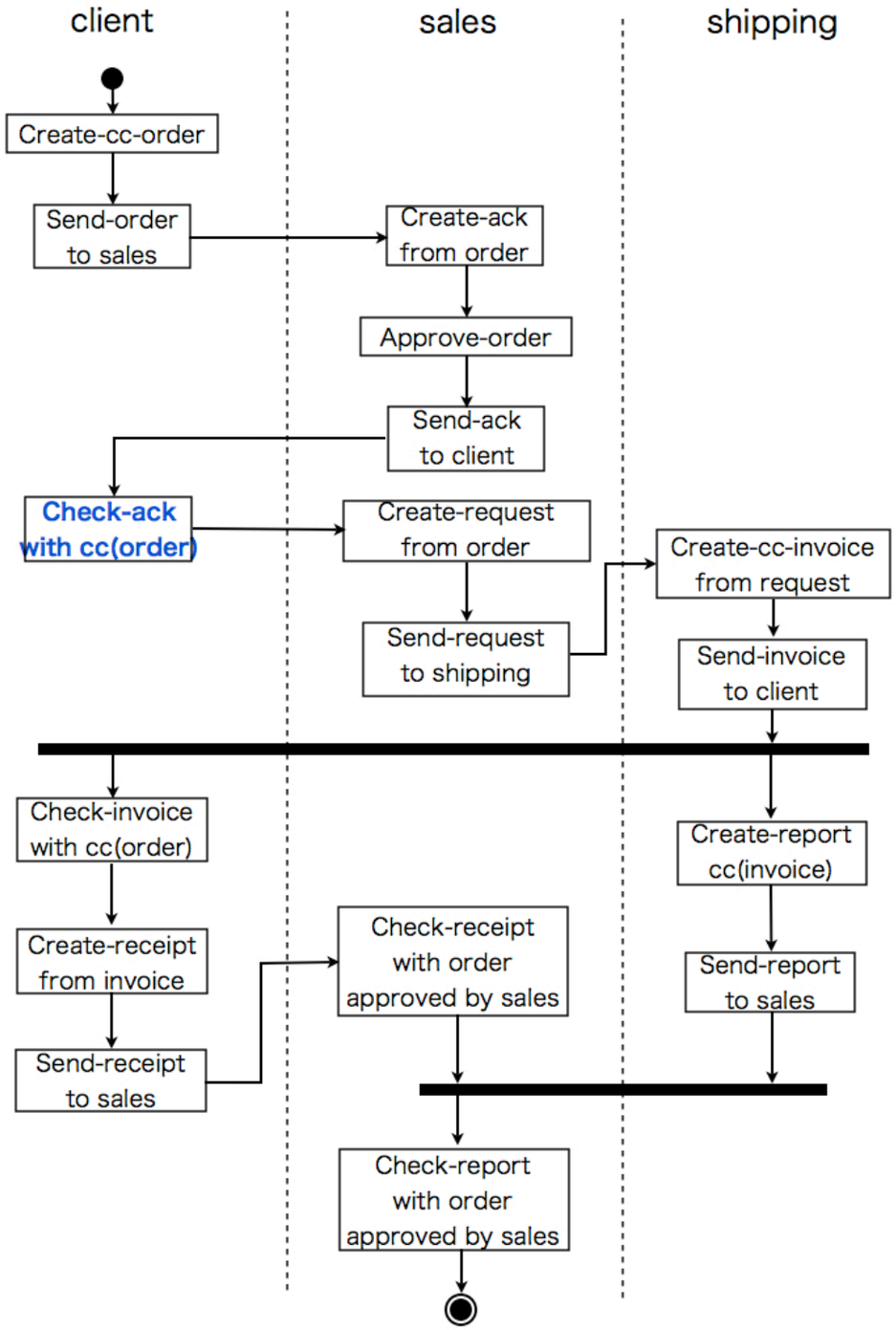
Example: Sales and Ship Process

- Example of checking a document
  - Checking ack with cc(order)

    Check-ack : State × DocumentID × DocumentID
    → State
    - transition rule

      Check-ack(S, D1, D2)
      = Check-1(S, client, D1, D2)
      if c-Check-ack(S, D1, D2)
      Check-ack(S, D1, D2) = S
      if not(c-Check-ack(S, D1, D2))

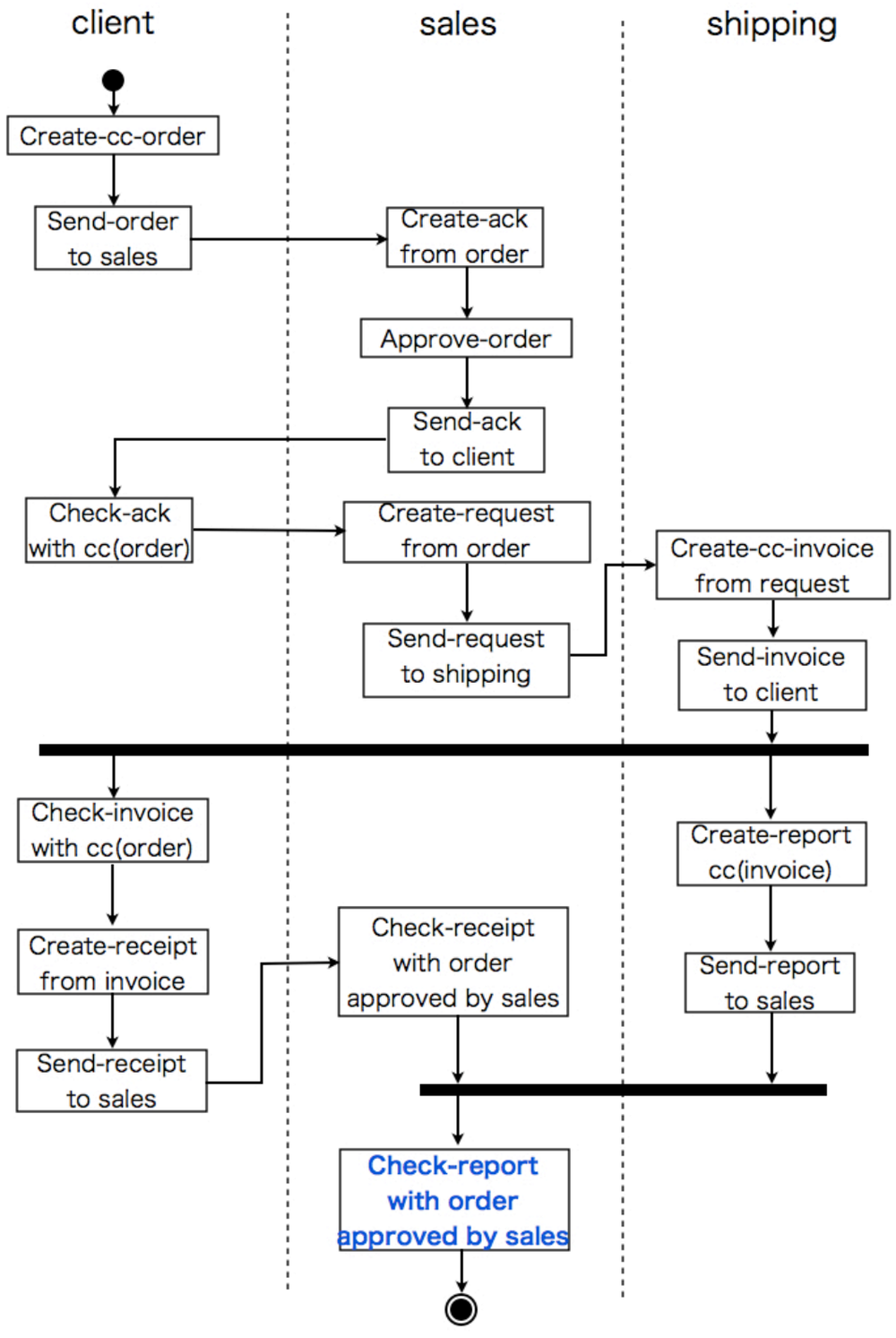
- c-Check-ack(S, D1, D2) is defined as follows
  - Type of D1 is ack
  - Type of D2 is cc(order)
  - ack is in client
  - cc(order) is in client



Property to be Proved for Controlling Risk in Single Sessions

- For Sales and Ship process,
  - If a report is in sales and has an evidence which shows it has been checked with order, the report is not forged.

inv1(S) =
((Place(S, report) = sales) and
 in?(ch(order), Evidences(S, report)))
implies
(Legal?(S, report) = true) .



Proof by Induction

- Initial states
```
op init : -> SaleAndShip
 -- for single session
  eq (I1 = I2) = true .
  eq Place(init, D) = noDivision .
  eq Evidences(init, D) = emptyE .
  eq SessionID(init, D) = noSessionID .
  eq OriginalSession(init, D) = noSessionID .
  eq Legal?(init, D) = true .
  eq DocumentIDList(init, I, T) = nilDID .
  eq UntrustedSet(init) = (sales shipping) .
```
- base case
```
  ops d : -> DocumentID .
  red inv1(init, d) .
```
- induction step

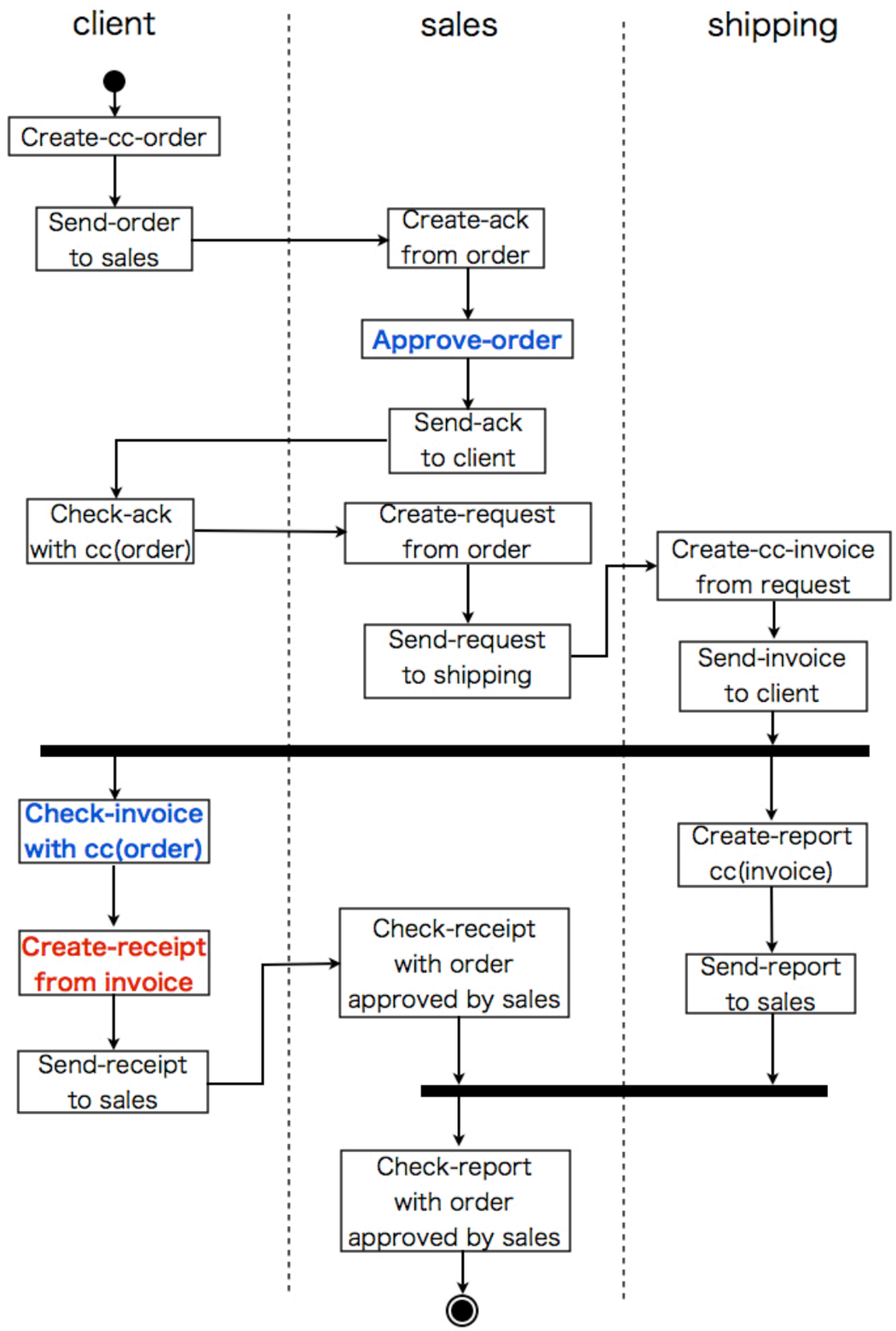
for Check-report
```
-- arbitrary objects
  ops s s' : -> SaleAndShip .
  ops d1 d2 d3 d4 : -> DocumentID .
  ops h1 h2 : -> EvidenceHistory .
  op dlist : -> DocumentIDList .
-- assumption
  -- eq c-Check-report(s, d1, d2) = true .
  eq Place(s, d1) = sales .
  eq Place(s, d2) = sales .
  eq SessionID(s, d2) = SessionID(s, d1) .
  eq DocumentType(s, d1) = report .
  eq DocumentType(s, d2) = order .
  eq DocumentIDList(s, SessionID(s, d1), receipt)
    = (d4 ; dlist) .
  eq d3 = d1 .
  -- eq in?(apv(sales), Evidences(s, d2)) = true .
  eq Evidences(s, d2) = (apv(sales) h2) .
  eq Legal?(s, d1) = Legal?(s, d2) .
  eq Legal?(s, d2) = false .
  eq in?(ch(order), Evidences(s, d1)) = false .
-- successor state
  eq s' = Check-report(s, d1, d2) .
-- check
  red (doc-inv(s, d4, d1, receipt) and inv2(s, d2, d4))
      implies
      (inv1(s, d3) implies inv1(s', d3)) .
```

- report is in sales
- order is in sales
- receipt is checked by order
- order has an evidence of approval
- order is forged
- boolean values of authenticity of order and report are the same
- report is not checked yet

Lemma

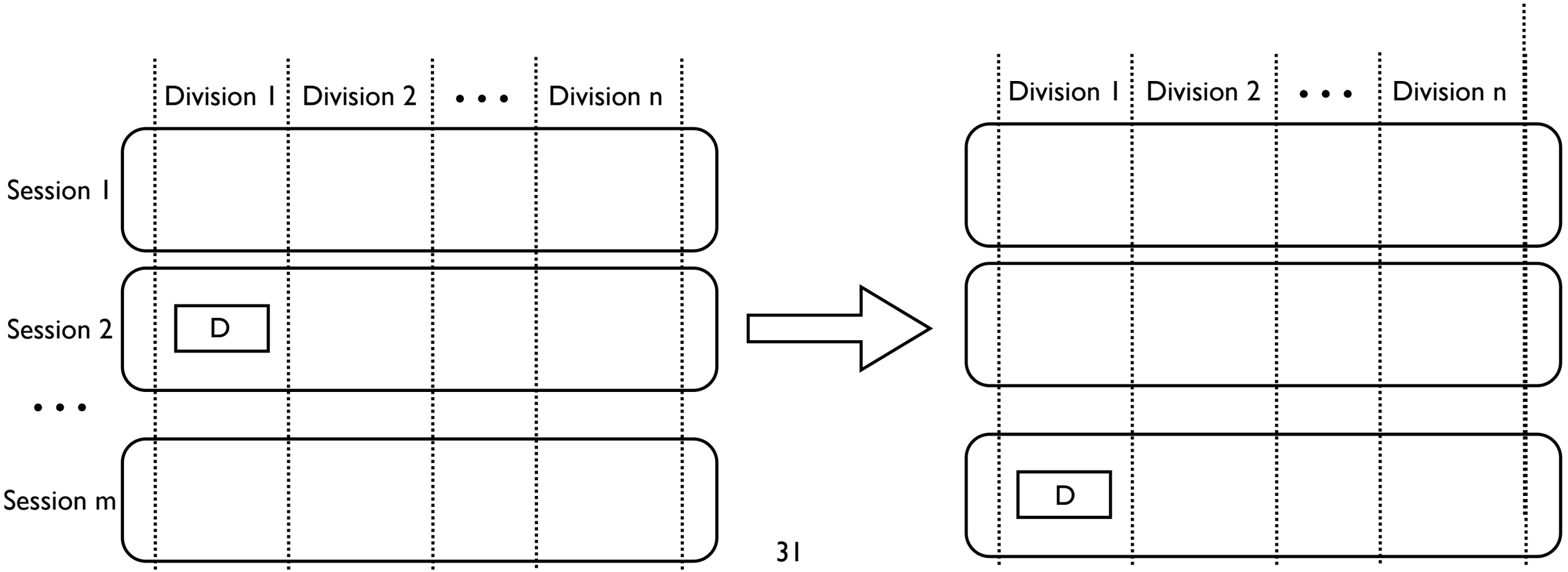
```
op inv2 : State DocumentID DocumentID -> Bool
var S : State
vars D1 D2 : DocumentID

-- if order has an evidence and is forged,
-- receipt will be never created
eq inv2(S, D1, D2)
     = ((DocumentType(S, D1) = order) and
        in?(apv(sales), Evidences(S, D1)) and
        (Legal?(S, D1) = false))
       implies
       not(DocumentType(S, D2) = receipt) .
```

- 9 lemmas are needed to prove the property



Irregular Event on Multiple Sessions

- Moving a document to another session
  ChangeSession :
  State × DocumentID × SessionID → State

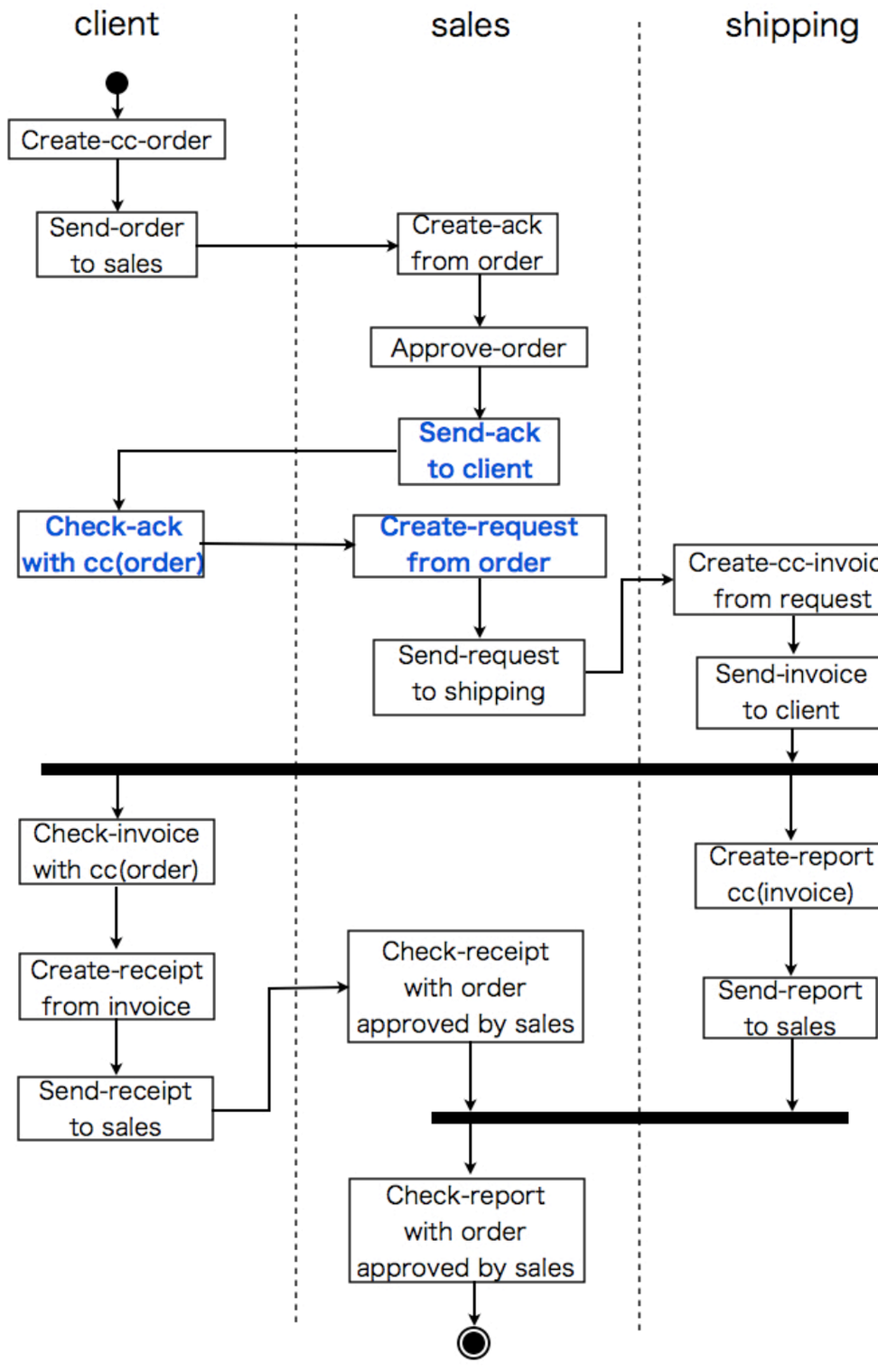


Example: Sales and Ship Process

- Example of moving a document to another session
  - moving a document in sales and ship process
    ChangeSession-SaleAndShip :
    State × DocumentID × SessionID → State
    - transition rule
      ChangeSession-SaleAndShip(S, D, I)
      = ChangeSession(S, D, I)
        if c-ChangeSession-SaleAndShip(S, D, I)
      ChangeSession-SaleAndShip(S, D, I) = S
      if not(c-ChangeSession-SaleAndShip(S, D, I))
- c-ChangeSession-SaleAndShip(S, D, I) is defined as follows
  - D is in an untrusted division

Influence of Moving a Document to Another Session

- Losing a document
  - For example, order in session 1 is moved to session 2 in early phase of the flow, checking documents with order cannot be done.
- Skipping a event
  - For example, if the ack is moved to session 2 after checking ack in session 1, request can be created in session 2 even if ack is not checked in session2.
- Receiving a different order
  - in the above case, client gets different products



Property to be Proved for Multiple Sessions

- Session IDs of Documents are the same as the Original Session IDs of them

inv-multi(S, D1, D2)
 = (*condition* and(SessionID(S, D1) = SessionID(S, D2)))
  implies
  (OriginalSessionID(S, D1) = OriginalSessionID(S, D2))

Conclusion and Future Work

- Conclusion
  - Formalization of business processes based on document logic
  - Formalization of controls and a risk on single sessions
  - Formalization of a risk on multiple sessions
  - Verification of effectivity of controls for single sessions
- Future Work
  - Formalization of controls on multiple sessions and verification
  - Tackling other examples