Evaluation of Control Activities in Business Processes by Formal Verification

ARIMOTO, Yasuhito
School of Information Science,
Japan Advanced Institute of Science and Technology

Control Activities

The policies and procedures that help ensure management directives are carried out

- by the Committee of Sponsoring Organizations of the Treadway Commission (COSO)

- They help ensure that necessary actions are taken to address risks.
- They include activities such as approvals, authorization, reconciliations and segregation of duties.

Evaluation of Control Activities

Are control activities in business processes are enough to deal with risks?

- Documents for evaluation of control activities in each business (example)
  - Flowchart of business activities:
    Representation of workflow
  - Risk and Control Matrix (RCM):
    Relations between risks and control activities
  - Business process narrative:
    Detailed description of the workflow

Our aim: Proposing a method for supporting evaluation of control activities in business processes by applying a formal method

Why Applying Formal Method?

- Formal description
  - excludes ambiguity and inconsistency
  - can be verified formally
- Formal verification
  - ensure the consistent evaluation of control activities
  - enables us to do exhaustive evaluation
  - gives scientific discussion on the result of the evaluation
    - why control activities are enough or not, etc.

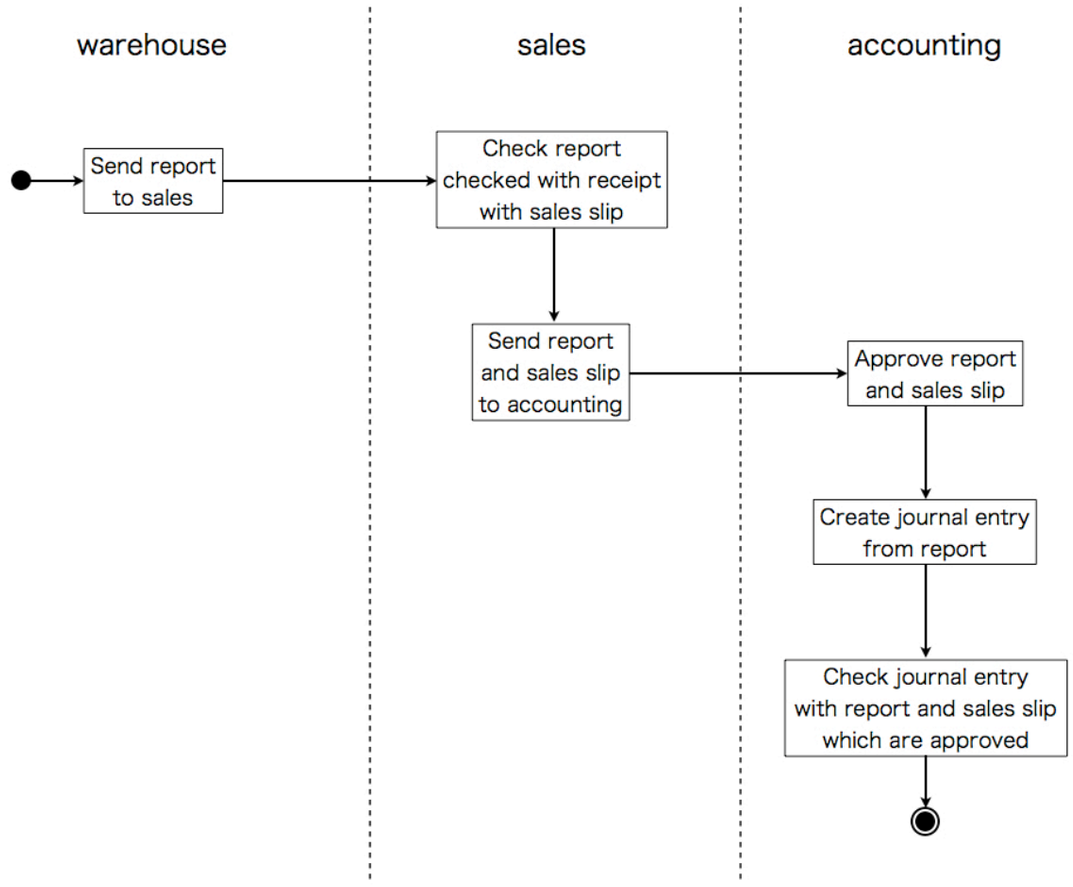
Evaluation by formal method gives us precise analysis of effectiveness of control activities and scientific discussion on the result

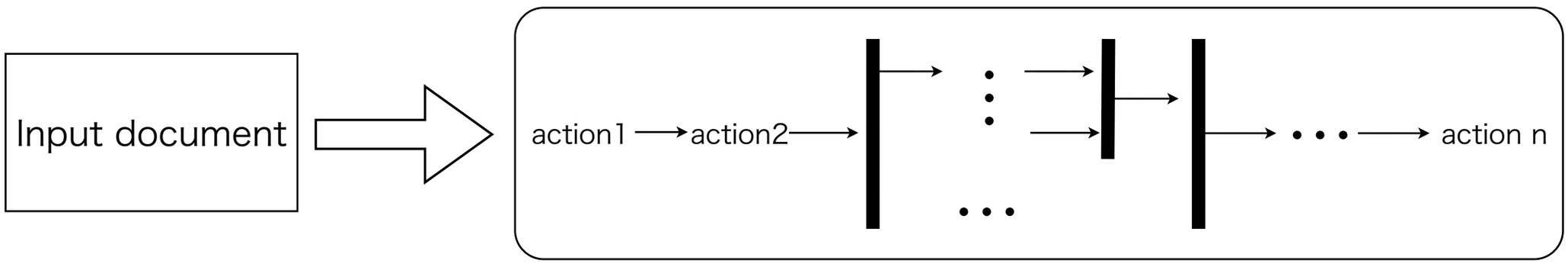
What are Formalized and Verified?

- Importance of documents
  - All information generated in business process are recorded in some documents
- Formal descriptions
  - Document flows in a business
  - Risks about falsification of documents
  - Control activities to avoid or detect falsification of documents
- Formal verification
  - Prove that control activities are enough for avoiding or detecting falsification of documents

Model of Document flows

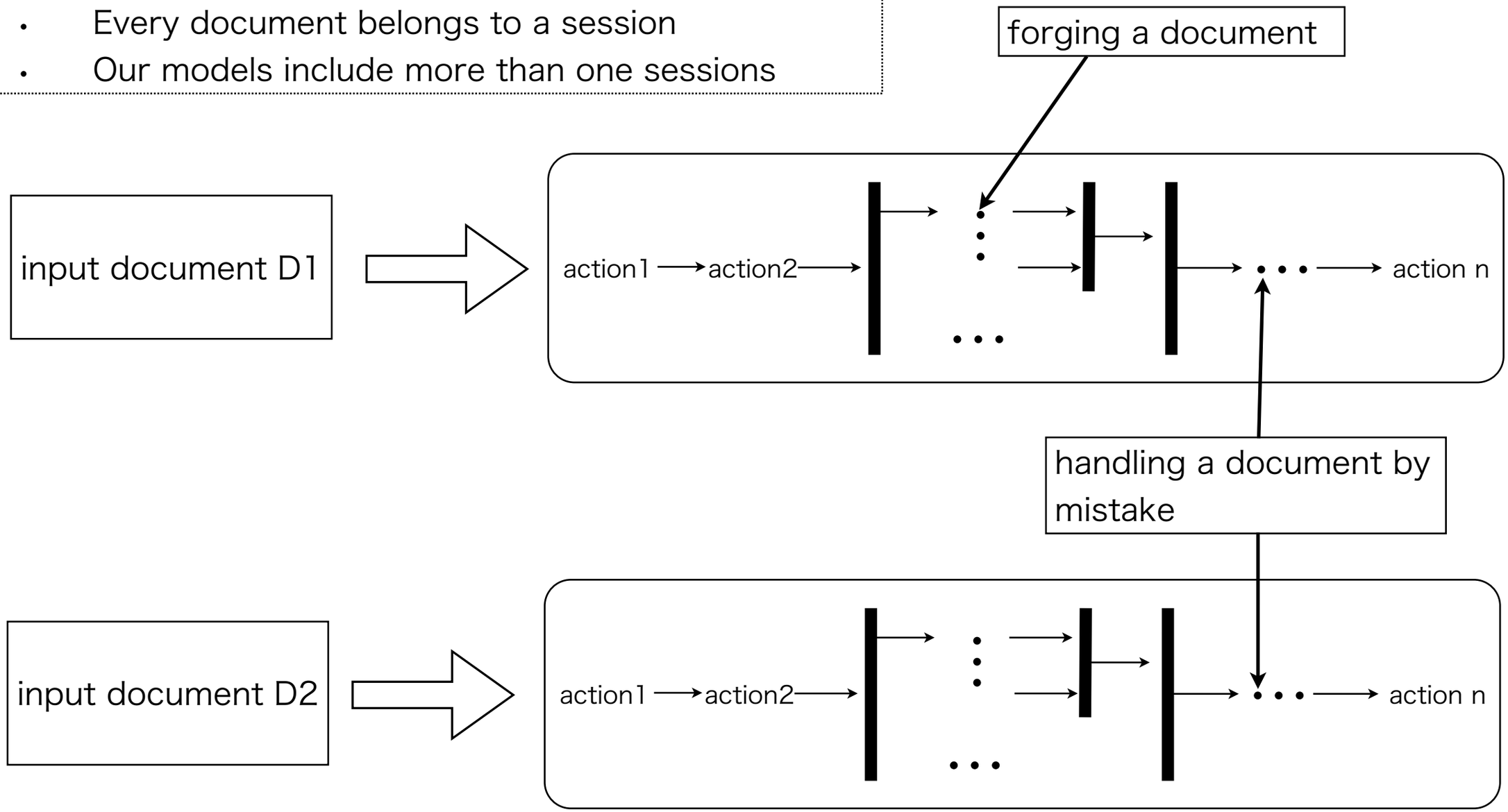
- Model of document flows has
  - input documents
    - e.g., report, sales slip
  - documents generated
    - e.g., journal entry
  - states which have achieved the goal
    - e.g., journal entry has been checked
  - actions
    - regular actions
      - e.g., send report to sales, etc.
    - irregular actions
      - e.g., forging a report, etc.
    - control actions
      - e.g., approving report, etc.





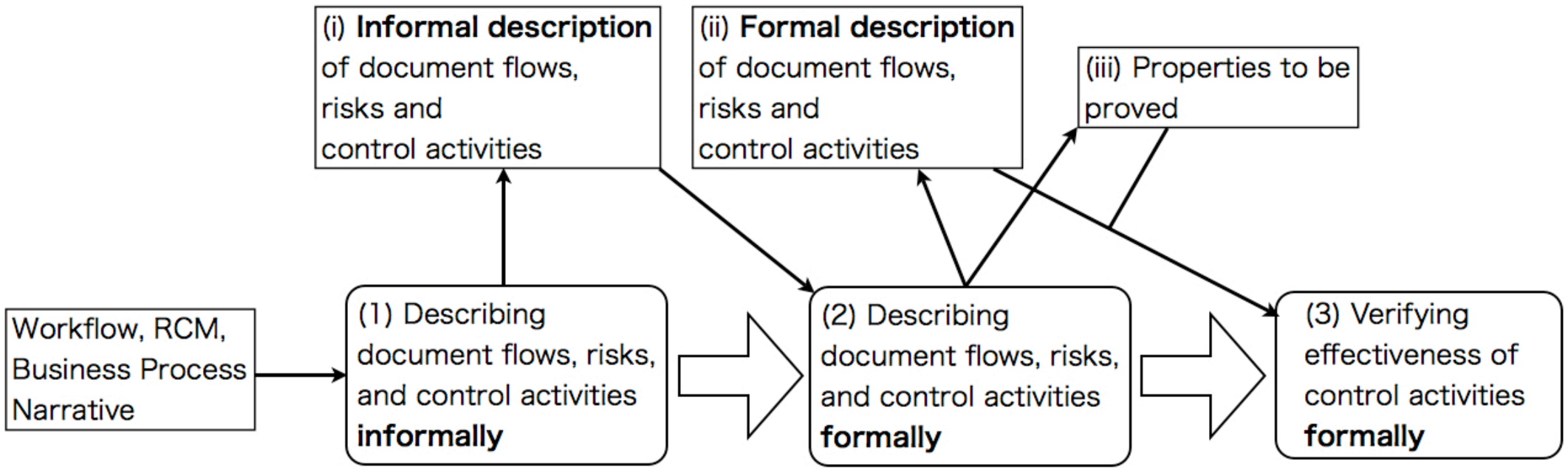
Influence by Irregular Actions

- A set of sequences of actions from specific input documents is called a session
- Every document belongs to a session
- Our models include more than one sessions

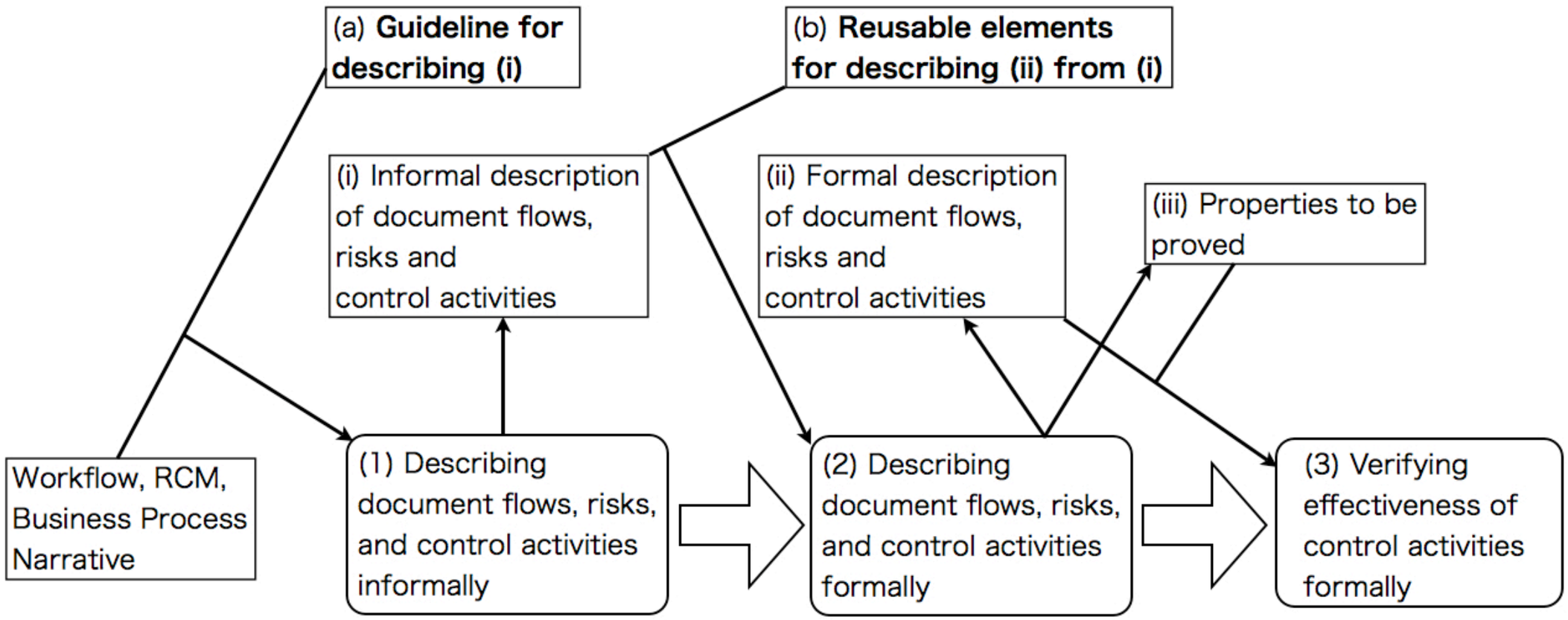


Procedure for Evaluation

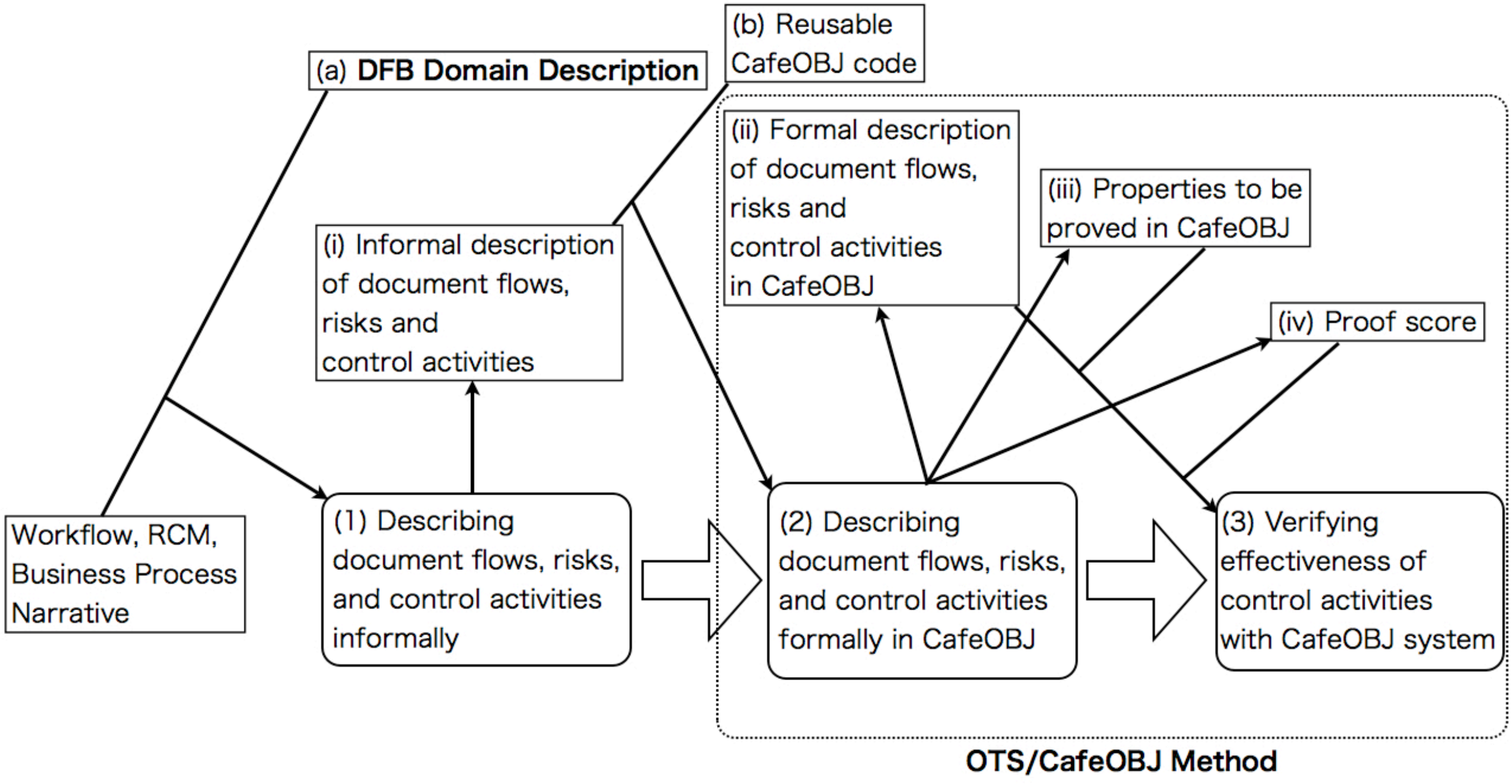
- Goal: Formal verification of effectiveness of control activities
  - For formal verification, formal description is necessary
  - For formal description, we must understand the objects to be formalized



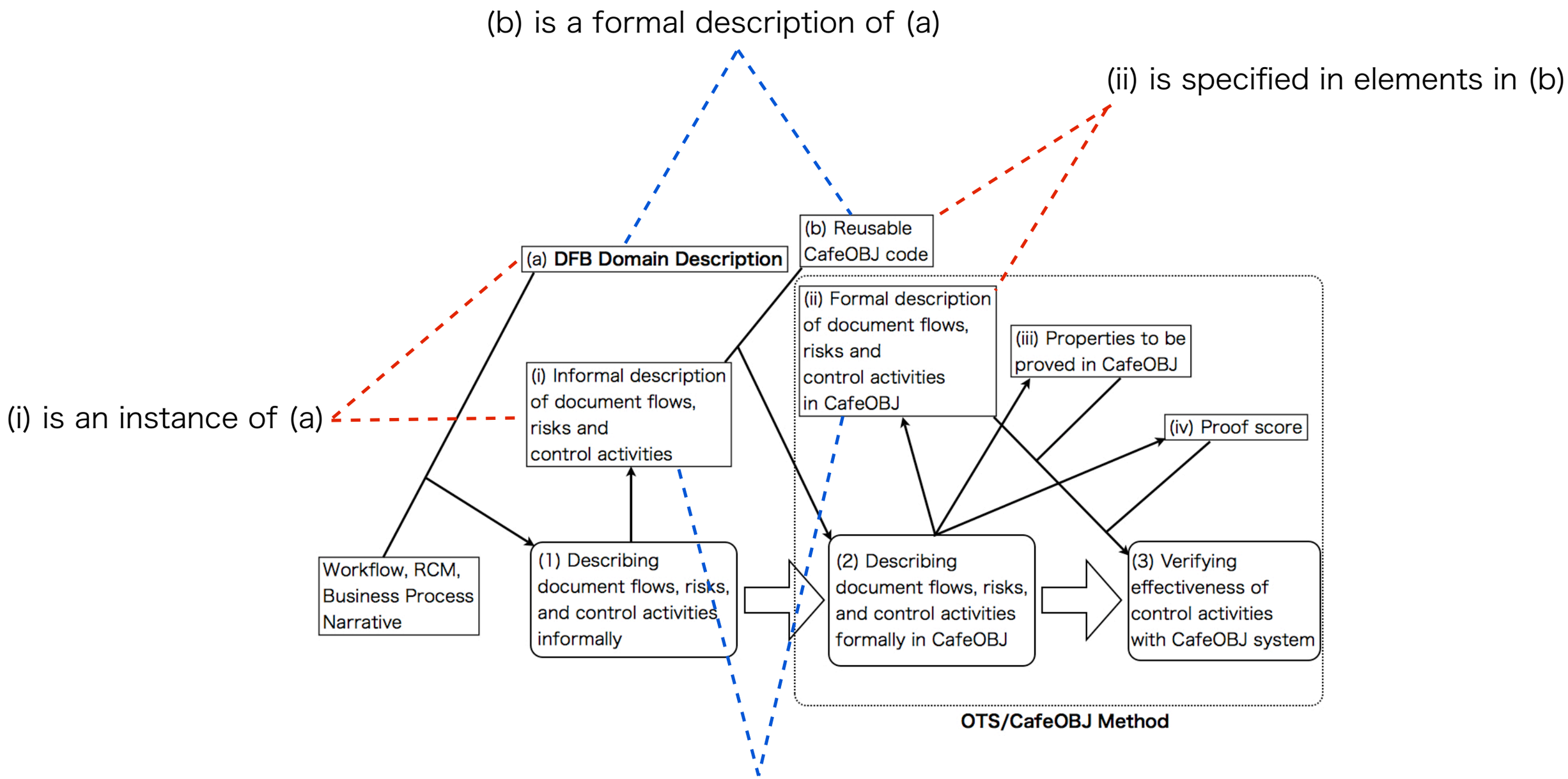
Support for Describing Document Flows

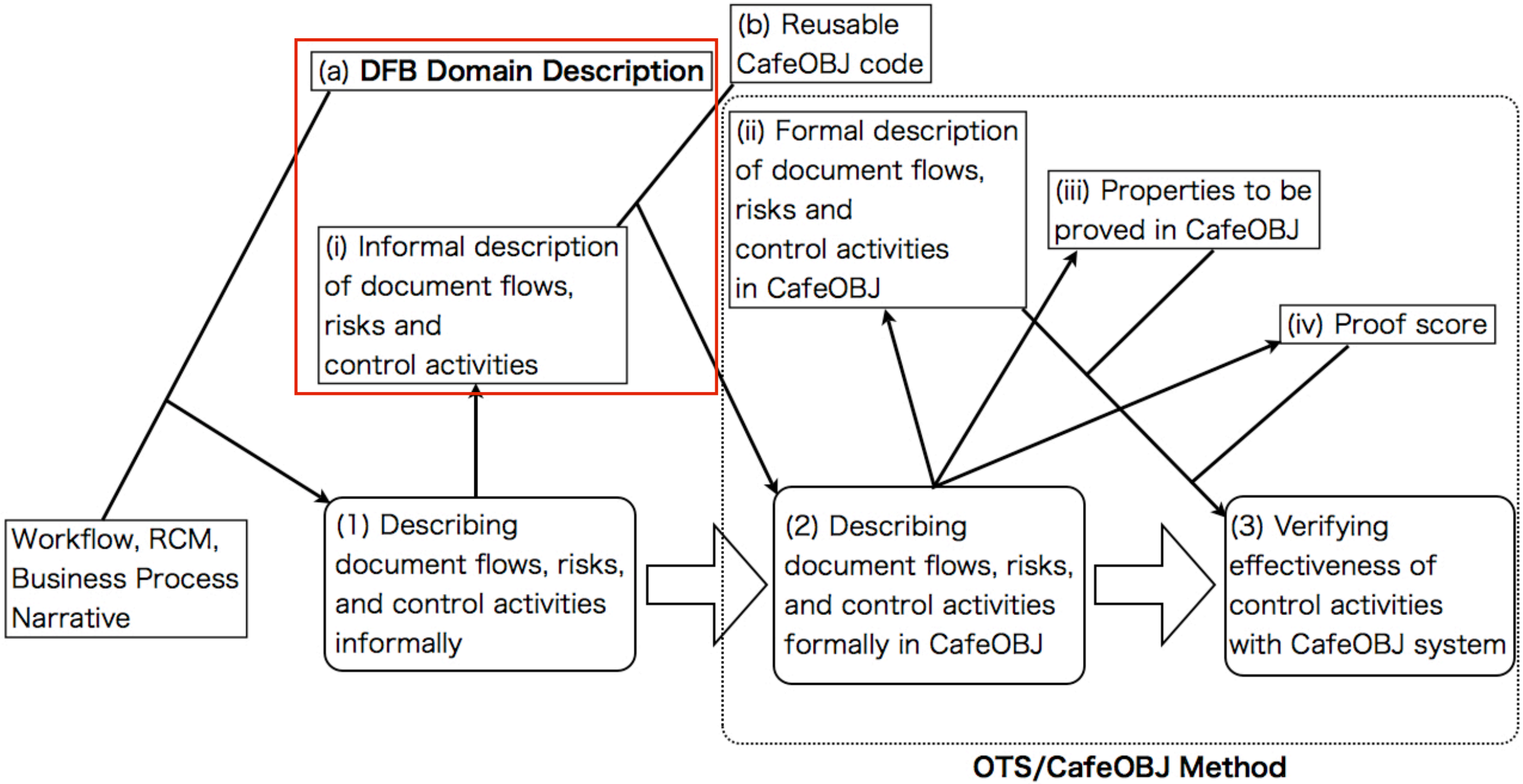


Approach


(a) DFB Domain Description
(b) Reusable CafeOBJ code
(i) Informal description of document flows, risks and control activities
(ii) Formal description of document flows, risks and control activities in CafeOBJ
(iii) Properties to be proved in CafeOBJ
(iv) Proof score
Workflow, RCM, Business Process Narrative
(1) Describing document flows, risks, and control activities informally
(2) Describing document flows, risks, and control activities formally in CafeOBJ
(3) Verifying effectiveness of control activities with CafeOBJ system
OTS/CafeOBJ Method

Overview of our Approach

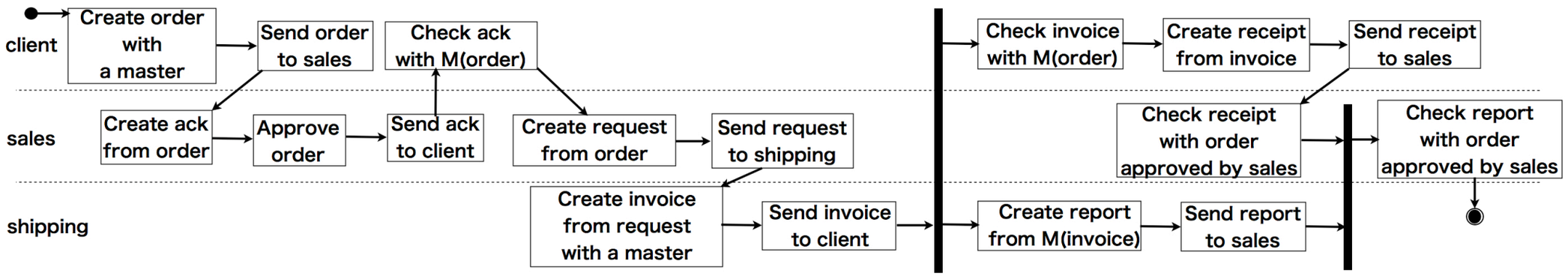



(a) DFB Domain Description
(b) Reusable CafeOBJ code
(i) Informal description of document flows, risks and control activities
(ii) Formal description of document flows, risks and control activities in CafeOBJ
(iii) Properties to be proved in CafeOBJ
(iv) Proof score
Workflow, RCM, Business Process Narrative
(1) Describing document flows, risks, and control activities informally
(2) Describing document flows, risks, and control activities formally in CafeOBJ
(3) Verifying effectiveness of control activities with CafeOBJ system
OTS/CafeOBJ Method

DFB Domain Description

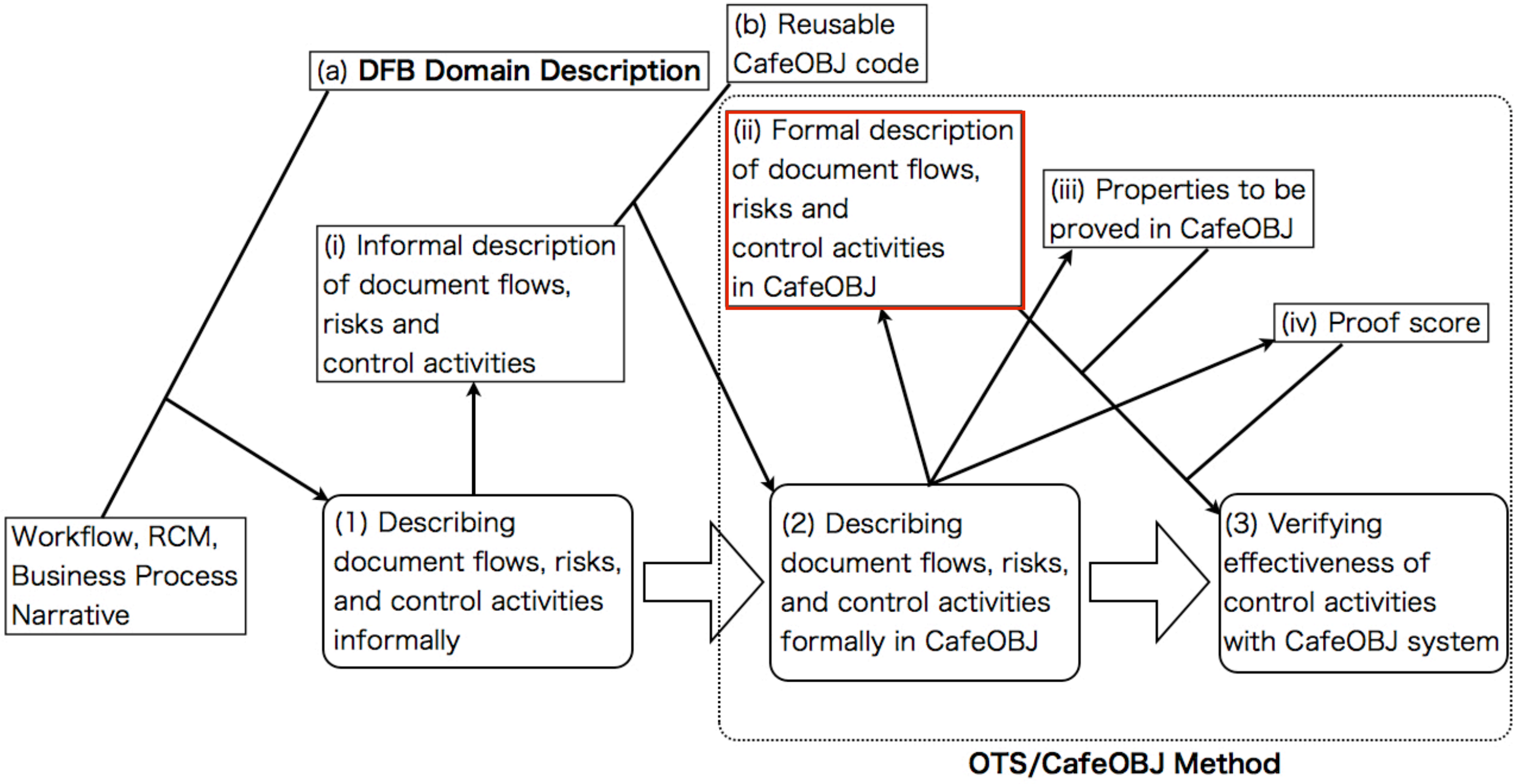
- A domain description (proposed by D. Bjorner)
  - is for understanding and analyzing a domain
  - consists of entities, functions, events, and behaviours of the domain
- DFB Domain (Domain of Document Flows in Business) :
  Domain constructed by activities on documents for business purpose
- Entities : Documents(attributes : document ID, document type, division name, etc.)
- Functions : taking documents, or attributes of documents and returns an attribute of a document
- Events
  - Regular events
    - Creating a document
    - Sending a document
  - Irregular events
    - Forging a document
    - Handling a document by mistake (a document is moved to another session)
  - Control events
    - Approving a document
    - Checking a document with another document
- A session is a set of behaviours which related to specific input documents

Example of Document Flow



sales and shipment process

- Documents : order, ack, request, etc.
- Events
  - Regular events
    - Creating order with a master, send order to sales, etc.
  - Irregular events
    - Forging a document, handling a document by mistake
  - Control events
    - Approving order, checking ack with the master document of order, etc.
- Final states for sessions : report is checked with order

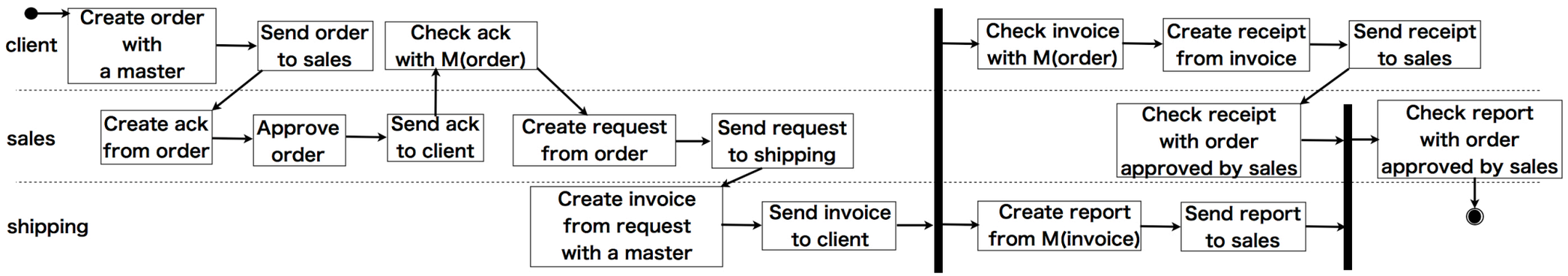

(a) DFB Domain Description
(b) Reusable CafeOBJ code
(i) Informal description of document flows, risks and control activities
(ii) Formal description of document flows, risks and control activities in CafeOBJ
(iii) Properties to be proved in CafeOBJ
(iv) Proof score
Workflow, RCM, Business Process Narrative
(1) Describing document flows, risks, and control activities informally
(2) Describing document flows, risks, and control activities formally in CafeOBJ
(3) Verifying effectiveness of control activities with CafeOBJ system
OTS/CafeOBJ Method

OTS/CafeOBJ Method

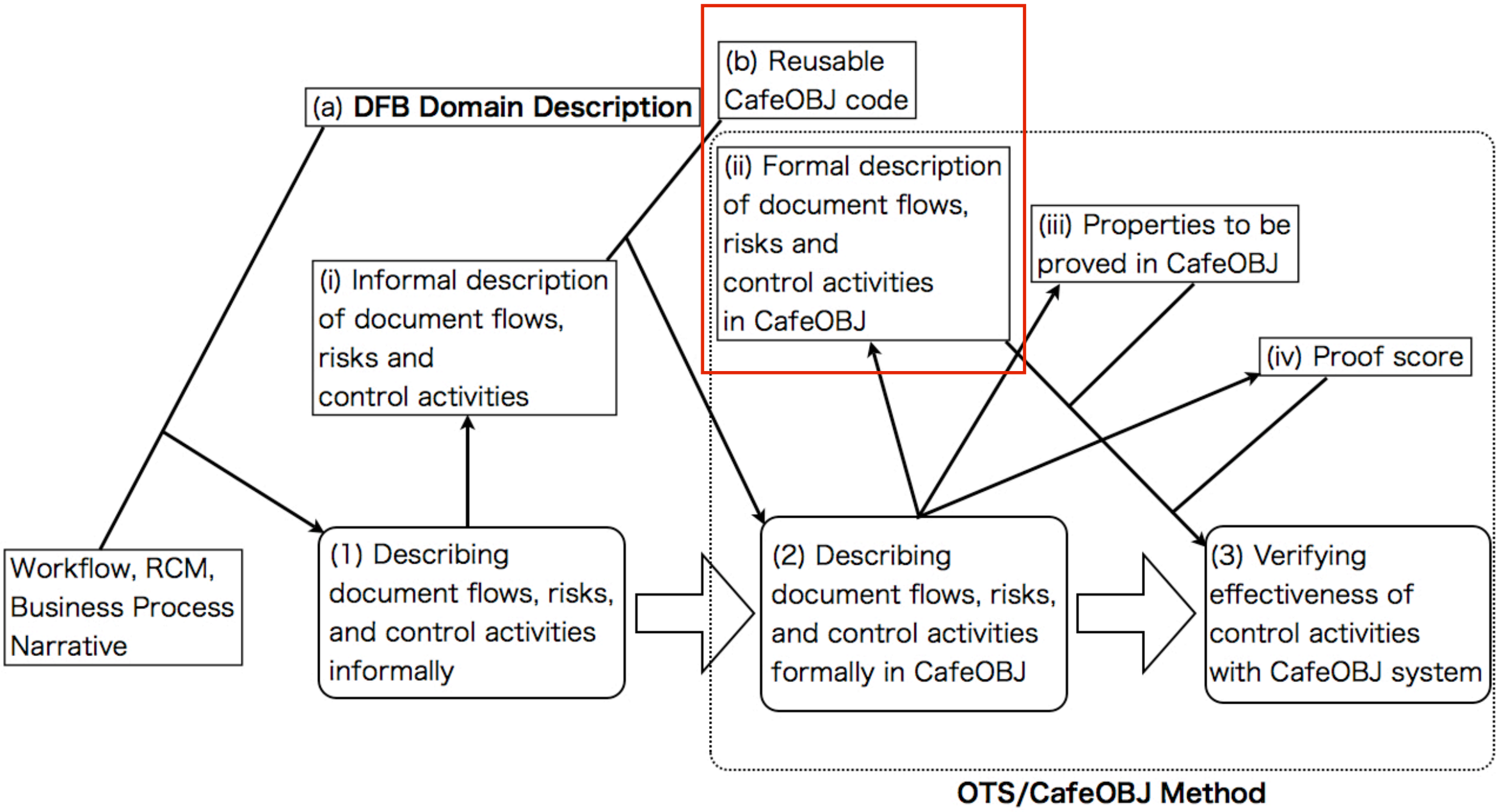
- A method for modeling, specifying and verifying behaviours of systems
  - Modeling behaviours of a system in OTS
  - Specifying OTS model in CafeOBJ
  - Proving properties by interaction between human and CafeOBJ system

- Observational Transition System (OTS) : defined as 3 tuples $\langle \mathcal{O}, \mathcal{I}, \mathcal{T} \rangle$
  - $\mathcal{O}$: A set of observations. States of system is characterized by obervations.
  - $\mathcal{I}$: A set of initial states
  - $\mathcal{T}$: A set of transitions. A state is changed by a transition if the effective condition is satisfied.

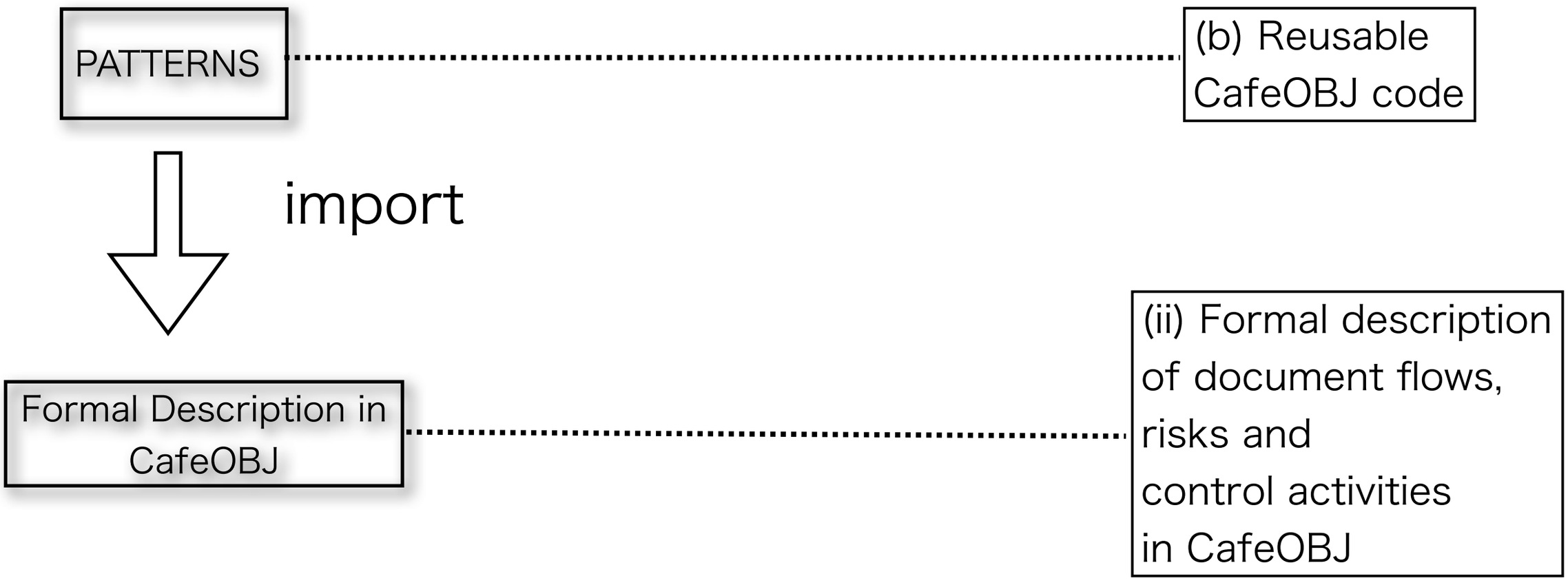
OTS Model of Sales and Shipment Process



- Documents are observed by observations
- Events are modeled as transitions
  - By events, values of documents are changed
  - The order of events can be specified in effective conditions
- In initial states, irregular events have not occurred yet.


(a) DFB Domain Description
(b) Reusable CafeOBJ code
(i) Informal description of document flows, risks and control activities
(ii) Formal description of document flows, risks and control activities in CafeOBJ
(iii) Properties to be proved in CafeOBJ
(iv) Proof score
Workflow, RCM, Business Process Narrative
(1) Describing document flows, risks, and control activities informally
(2) Describing document flows, risks, and control activities formally in CafeOBJ
(3) Verifying effectiveness of control activities with CafeOBJ system
OTS/CafeOBJ Method

Formal Description of Document Flows



Formal Description

PATTERN

```
mod PATTERNS {
  pr ( DIVISIONSET + EQBOOL + DOCUMENT +
       DOCUMENTIDSET )
  *[ State ]*

  bop Document : State DocumentID -> Document
  bop Legal? : State DocumentID -> Bool
  bop OriginalSessionID : State DocumentID -> SessionID
  bop DocumentIDSet : State SessionID DocumentType -> DocumentIDSet
  bop UntrustedSet : State -> DivisionSet
--
  var S : State  vars T T1 T2 : DocumentType  vars D D1 D2 D3 : DocumentID
  vars I I1 I2 : SessionID  var DL : DocumentIDSet
  vars T T1 T2 T3 : DocumentType vars V V1 V2 : Division
 vars E E1 E2 : Evidence
--
-- Create-1
  op c-Create-1 : State SessionID Division DocumentID DocumentType -> Bool
  eq c-Create-1(S, I, V, D, T) = (Document(S, D) = noDocument) .
--
  op Create-1 : State SessionID Division DocumentID DocumentType -> State
  eq Document(Create-1(S, I, V, D1, T), D2)
     = if (D1 = D2) then mkDocument(D1, T, V, emptyE, I)
       else Document(S, D2) fi .
  eq Legal?(Create-1(S, I, V, D1, T), D2)
     = if (D1 = D2) then true else Legal?(S, D2) fi .
  eq OriginalSessionID(Create-1(S, I, V, D1, T), D2)
     = if (D1 = D2) then I else OriginalSessionID(S, D2) fi .
  eq DocumentIDSet(Create-1(S, I1, V, D1, T1), I2, T2)
     = if (I1 = I2) and (T1 = T2) then (D1 DocumentIDSet(S, I1, T1))
       else DocumentIDSet(S, I2, T2) fi .
  eq UntrustedSet(Create-1(S, I, V, D, T)) = UntrustedSet(S) .
--
-- Create-2
  op c-Create-2 :
       State Division DocumentID DocumentType DocumentID -> Bool
  eq c-Create-2(S, V, D1, T, D2) =
      (Document(S, D1) = noDocument) and
      (getDivision(Document(S, D2)) = V) .
  op Create-2 :
       State Division DocumentID DocumentType DocumentID -> State
... ... ...

}
```

Definition of Observations

Definition of initial states

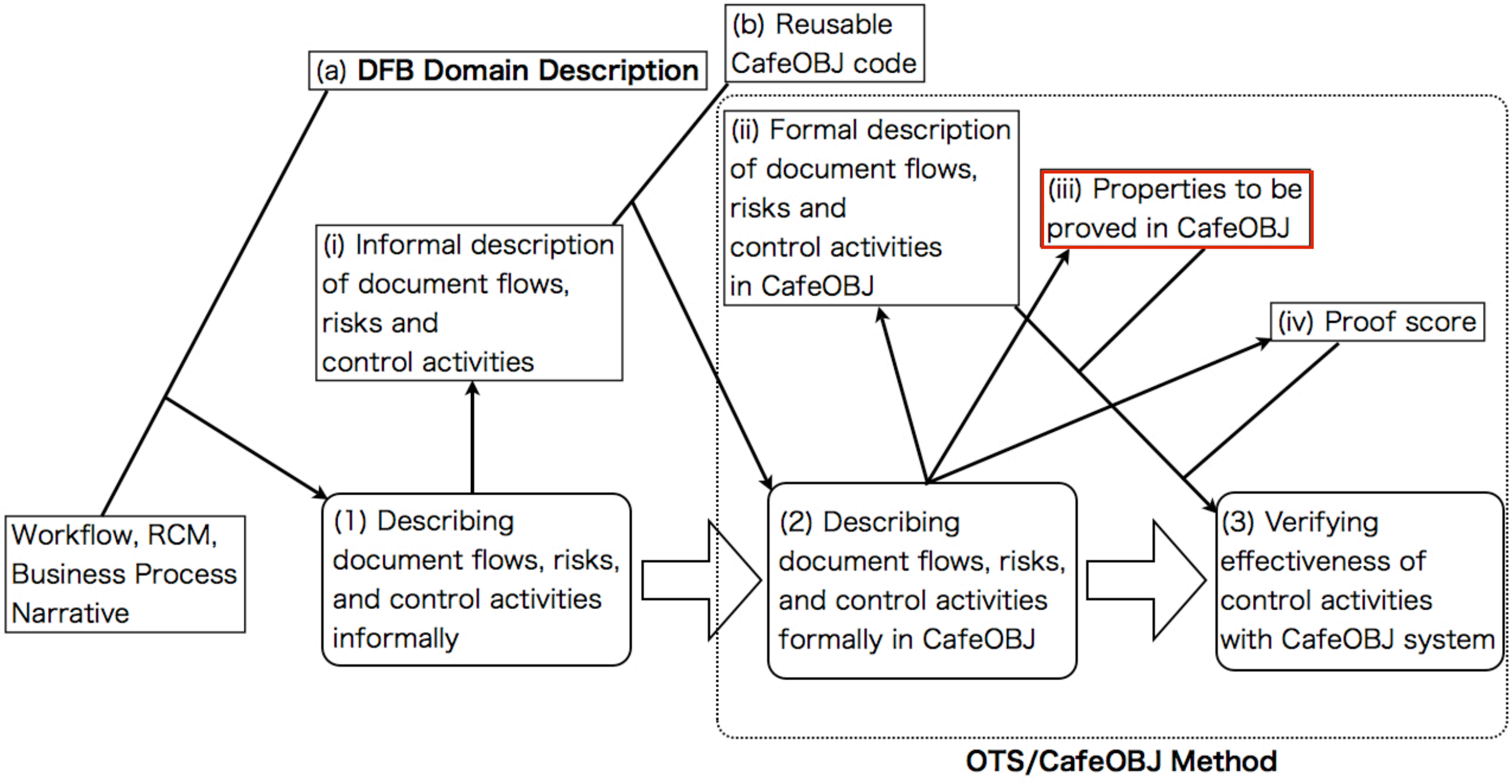
import

transition patterns, and framework of effective condition

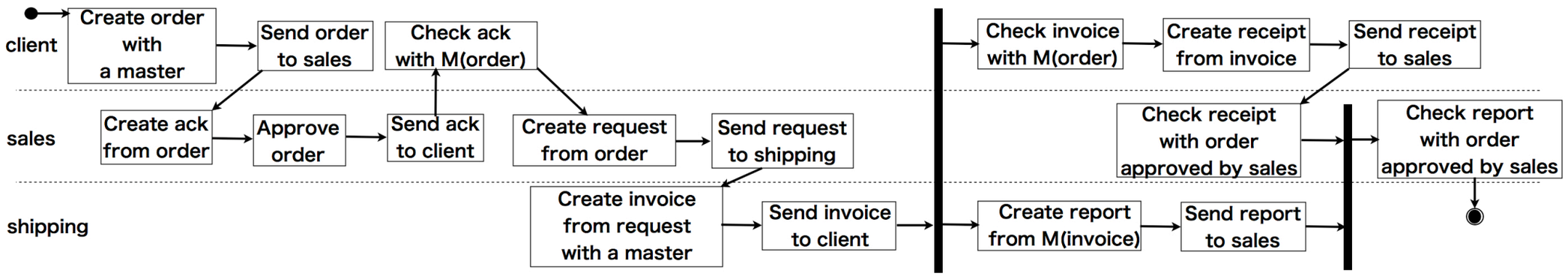
Definition of transition rules

Formal Description of document flows

```
mod* SALE-AND-SHIP {
  pr ( PATTERNS )
  *[ SaleAndShip < State ]*

  var S : SaleAndShip var T : DocumentType var V : Division vars D D1 D2 D3 :
DocumentID vars I I1 I2 : SessionID
--
  op init : -> SaleAndShip
  eq Document(init, D) = noDocument .   eq Legal?(init, D) = true .
  eq OriginalSessionID(init, D) = getSessionID(Document(init, D)) .
  eq DocumentIDSet(init, I, T) = emptyDID .  eq UntrustedSet(init) = (sales shipping) .
--
--  Create-cc-order
  op c-Create-cc-order : SaleAndShip SessionID DocumentID  -> Bool
  eq c-Create-cc-order(S, I, D)
      = c-Create-cc-1(S, I, client, D, order) and
      not(I = noSessionID) and (DocumentIDSet(S, I, order) = emptyDID) and
      (DocumentIDSet(S, I, cc(order)) = emptyDID) .
--
  bop Create-cc-order : SaleAndShip SessionID DocumentID  -> SaleAndShip
  ceq Create-cc-order(S, I, D) = Create-cc-1(S, I, client, D, order)
      if c-Create-cc-order(S, I, D) .
  ceq Create-cc-order(S, I, D) = S  if not(c-Create-cc-order(S, I, D)) .
--
-- Send-order
  op c-Send : SaleAndShip DocumentID -> SaleAndShip
     = c-Send(S, client, D, sales) and (getDocumentType(Document(S, D)) = order) .
--
  bop Send-order : SaleAndShip DocumentID -> SaleAndShip
  eq c-Send-order(S, D)
     = c-Send(S, client, D, sales) and
      (getDocumentType(Document(S, D)) = order) .
  ceq Send-order(S, D) = Send(S, client, D, sales) if c-Send-order(S, D) .
  ceq Send-order(S, D) = S if not(c-Send-order(S, D)) .
... ... ...

}
```


(a) DFB Domain Description
(b) Reusable CafeOBJ code
(i) Informal description of document flows, risks and control activities
(ii) Formal description of document flows, risks and control activities in CafeOBJ
(iii) Properties to be proved in CafeOBJ
(iv) Proof score
Workflow, RCM, Business Process Narrative
(1) Describing document flows, risks, and control activities informally
(2) Describing document flows, risks, and control activities formally in CafeOBJ
(3) Verifying effectiveness of control activities with CafeOBJ system
OTS/CafeOBJ Method

Properties to be Proved



sales and shipment process

Undesirable states :
one or more documents are forged or have been moved to another session

- States in which risk occurs

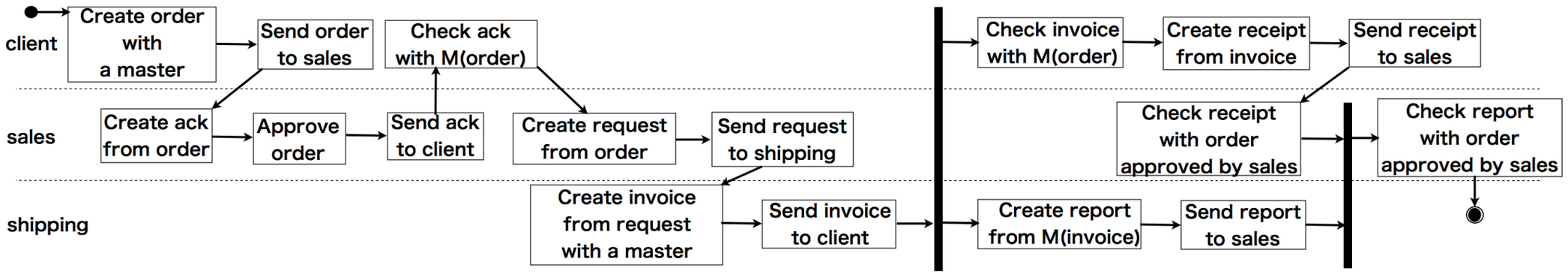
document D is report and
D is in sales and
D has been checked with order and
D is forged

- Property

(document D is report and
D is in sales and
D has been checked with order) implies
D is not forged

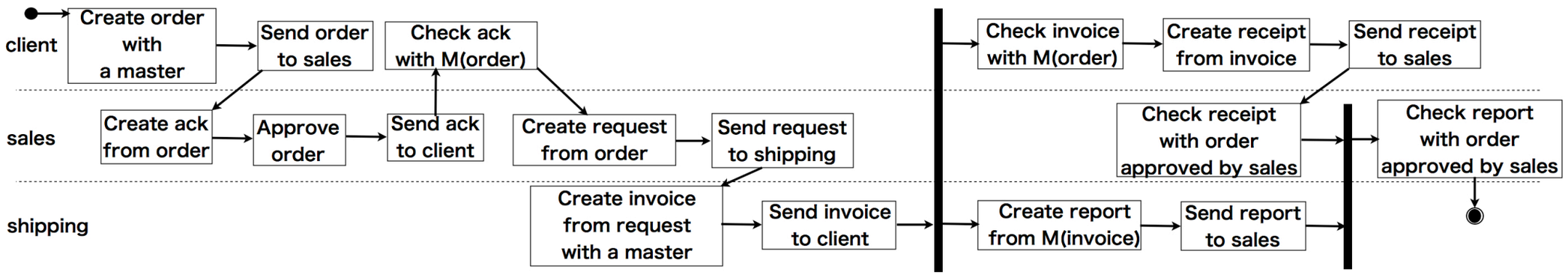
```
op inv1 : State DocumentID -> Bool

inv1(S, D)
= ((getDocumentType(Document(S, D)) = report) and
   (getDivision(Document(S, D)) = sales) and
   in?(ch(order), getEvidenceHistory(Document(S, D))))
  implies
  (Legal?(S, D) = true) .
```

Understanding the Problem

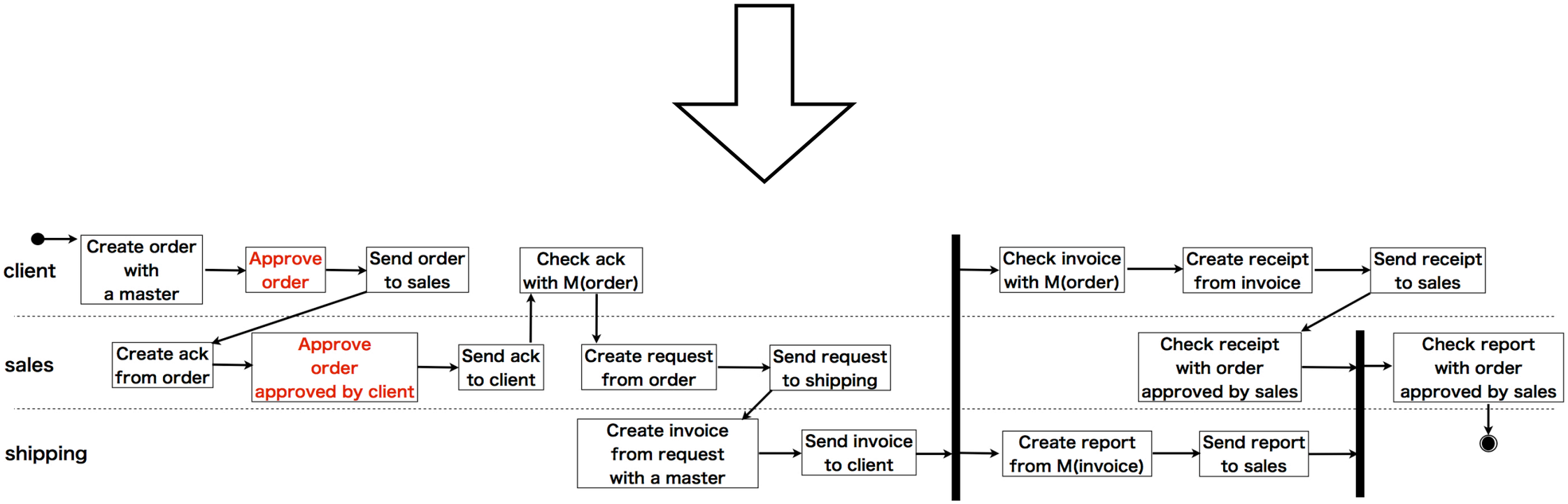


- By lemma discovery
  - What should be satisfied to prove the property
  - Through discovering lemma, we analyze why control activities are enough or not
- To prove inv1, inv2 and inv3 are needed
  - inv2 : if a document D1 is order, has been approved and has been forged, receipt document does not exist in the session
  - inv3 : if a document D is report and is in shipping division, D is not checked by order document.
- We need some more lemmas to prove inv2.

Result of Verification

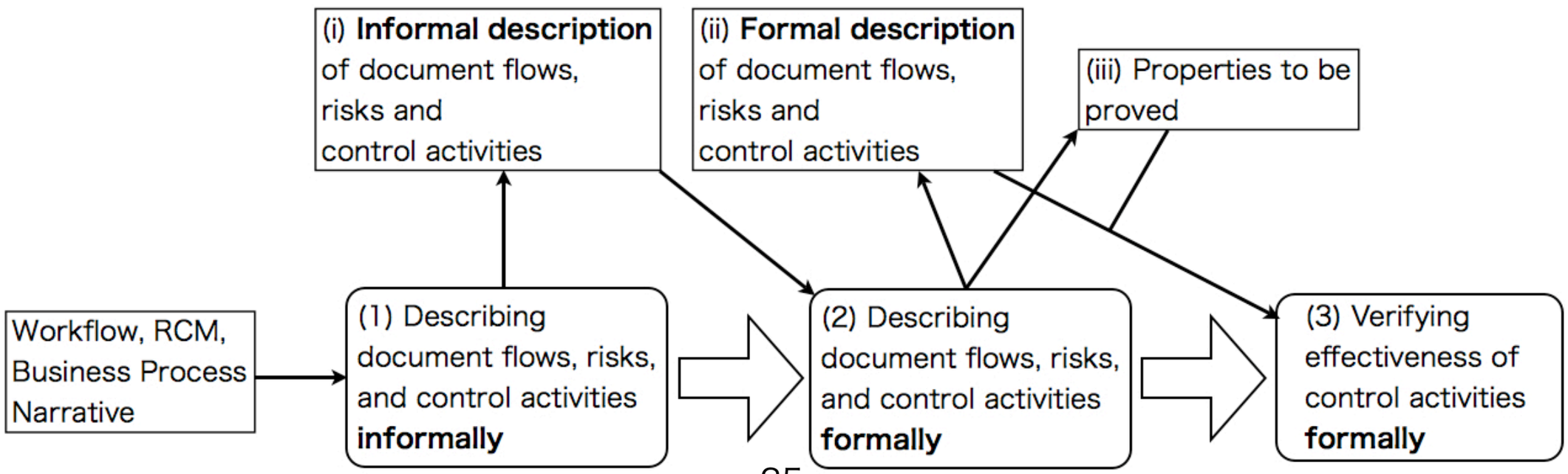


- Only one session：inv1 is proved
- More than one sessions：inv1 cannot be proved



Conclusion and Future Work

- Conclusion
  - Formalization of document flows, risks, and control activities
    - Development of DFB Domain Description
    - Development of reusable CafeOBJ code for the formal description
  - Formal analysis of effectiveness of control activities
    - Application of OTS/CafeOBJ Method to the evaluation
- Future Work
  - Evaluation of our method
    - Formalization of patterns of risks and control activities
    - More examples
    - Analysis of results of verification
  - Supporting verification
    - Analysis of patterns of lemmas, analysis of patterns of risks and control activities



Thank you