

# DiALog

## Distributed Auditing Logs

Adapt

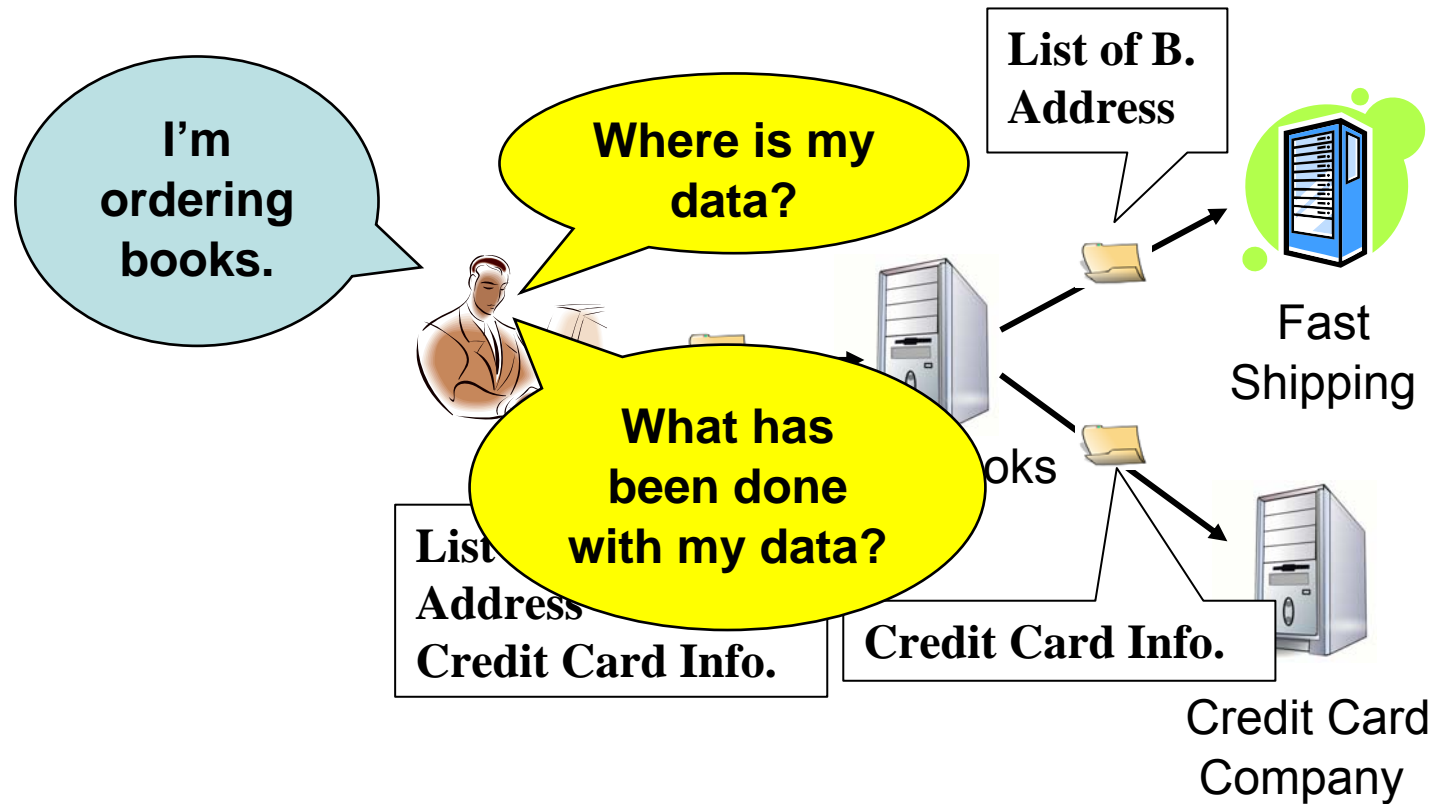
05.05.2009

***Christoph Ringelstein***

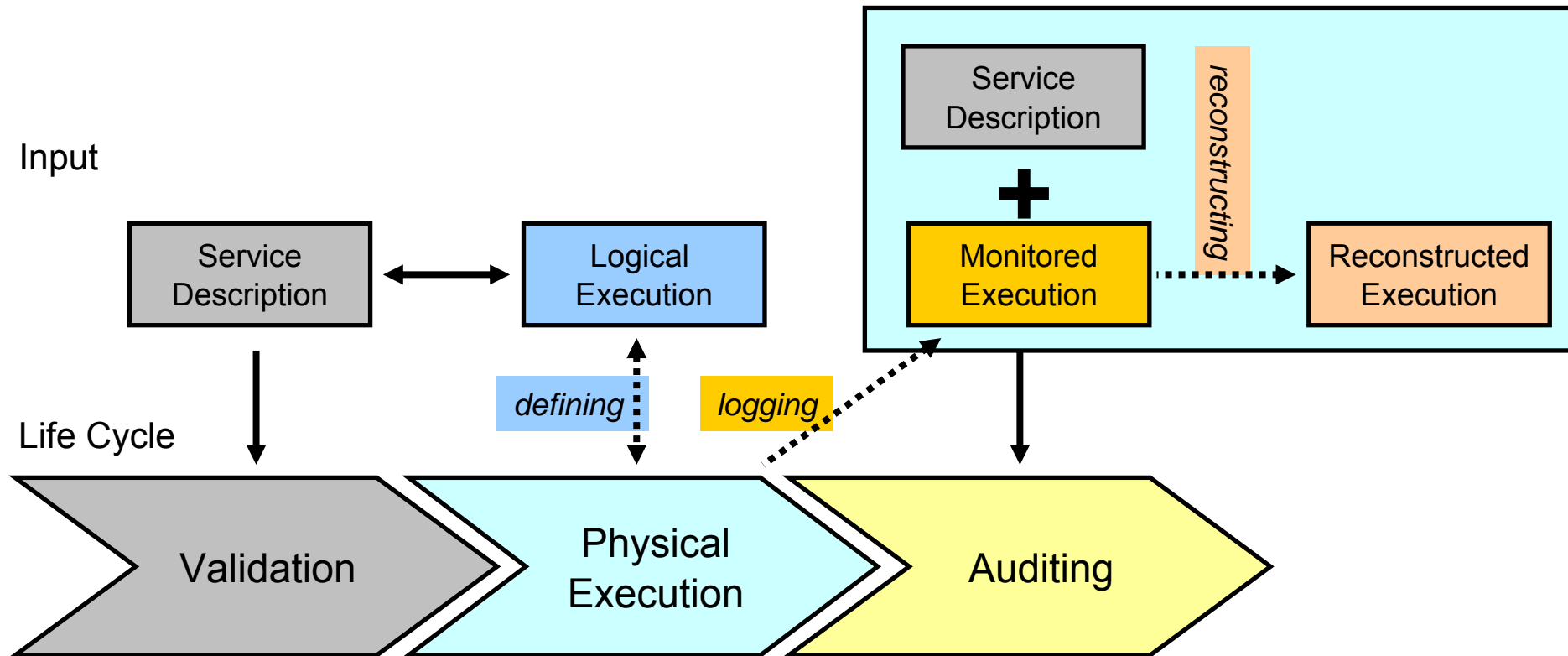


# **Auditing Distributed Data Processing**

## Distributed Data Processing

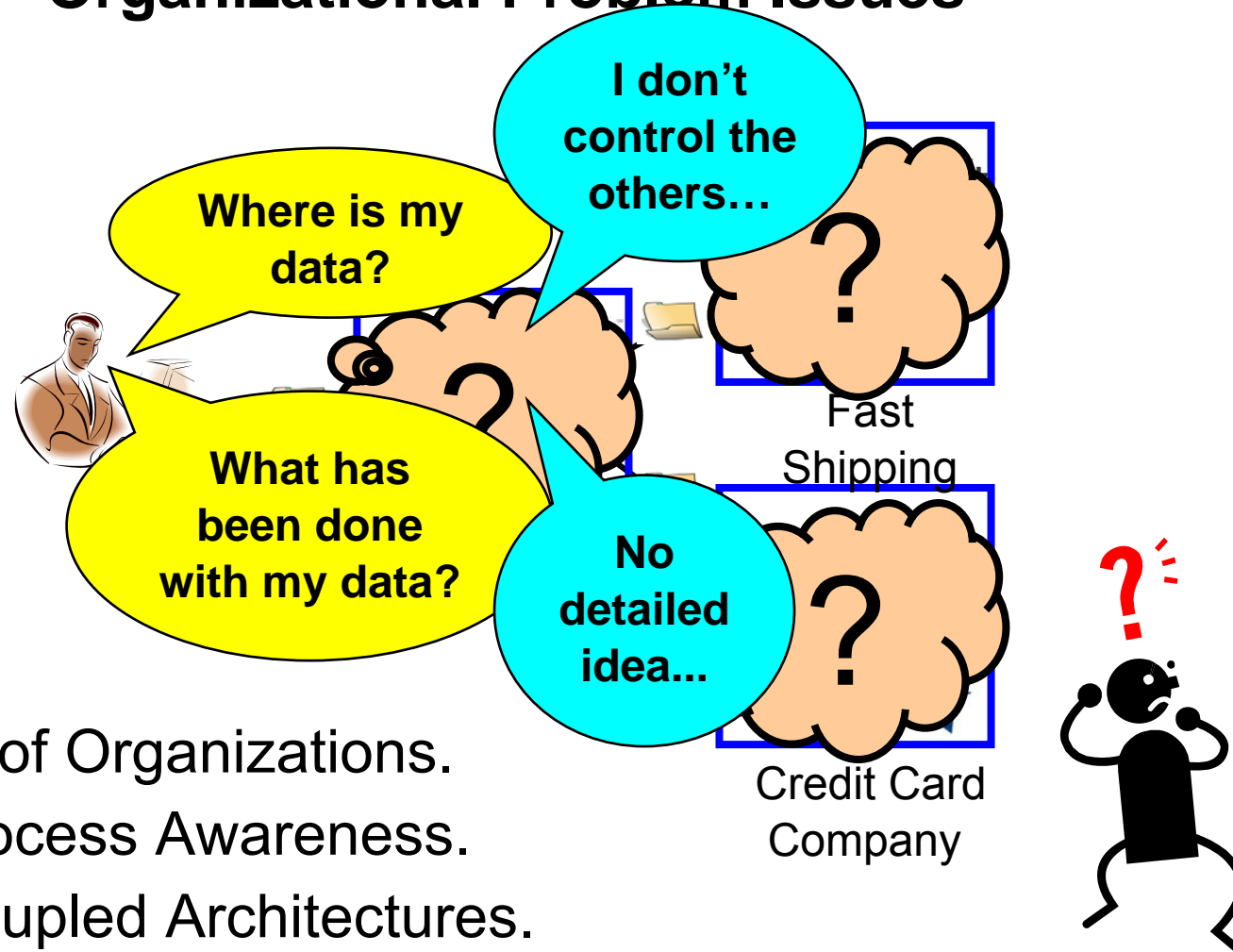


## Life Cycle of Controlling Distributed Data Processing



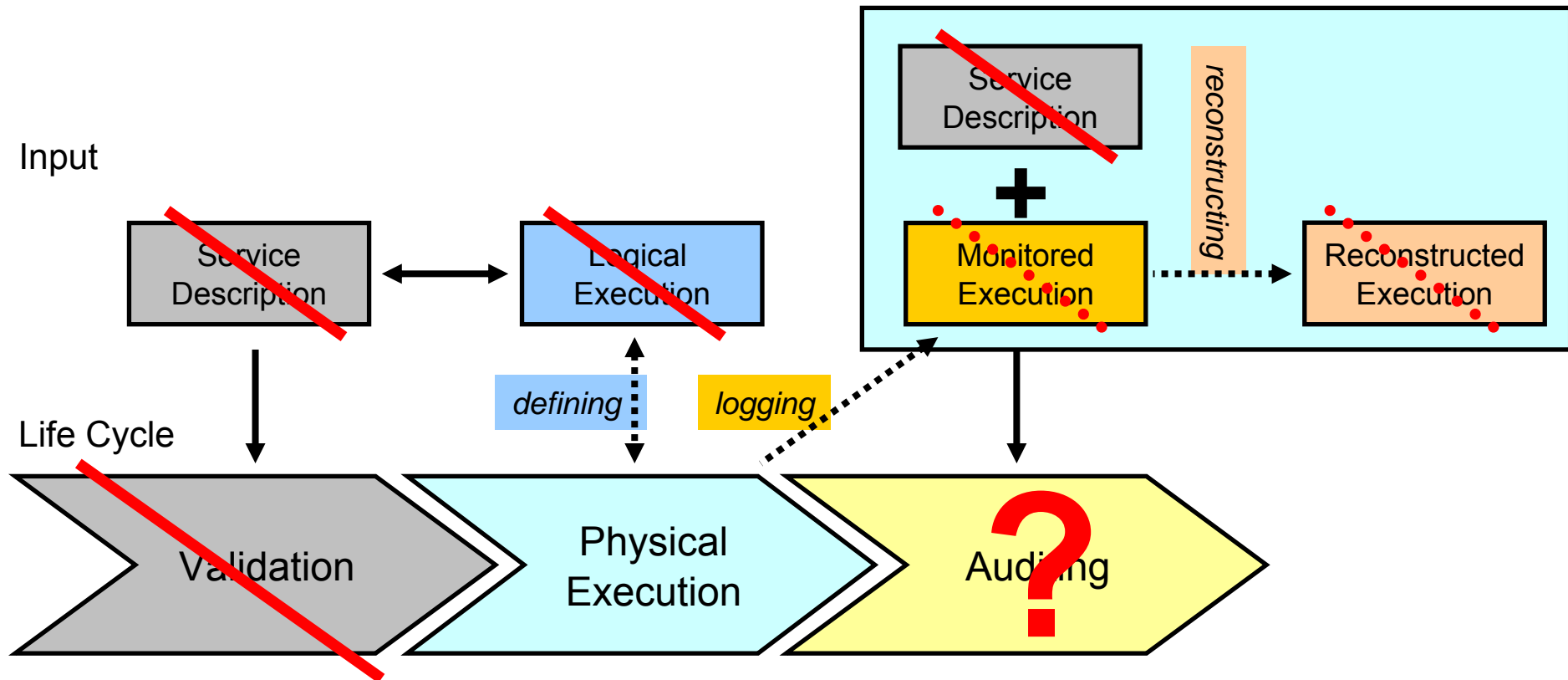
# The Problem

# Organizational Problem Issues



Autonomy of Organizations.  
Lack of Process Awareness.  
Loosely-coupled Architectures.

# Life Cycle of Controlling Distributed Data Processing



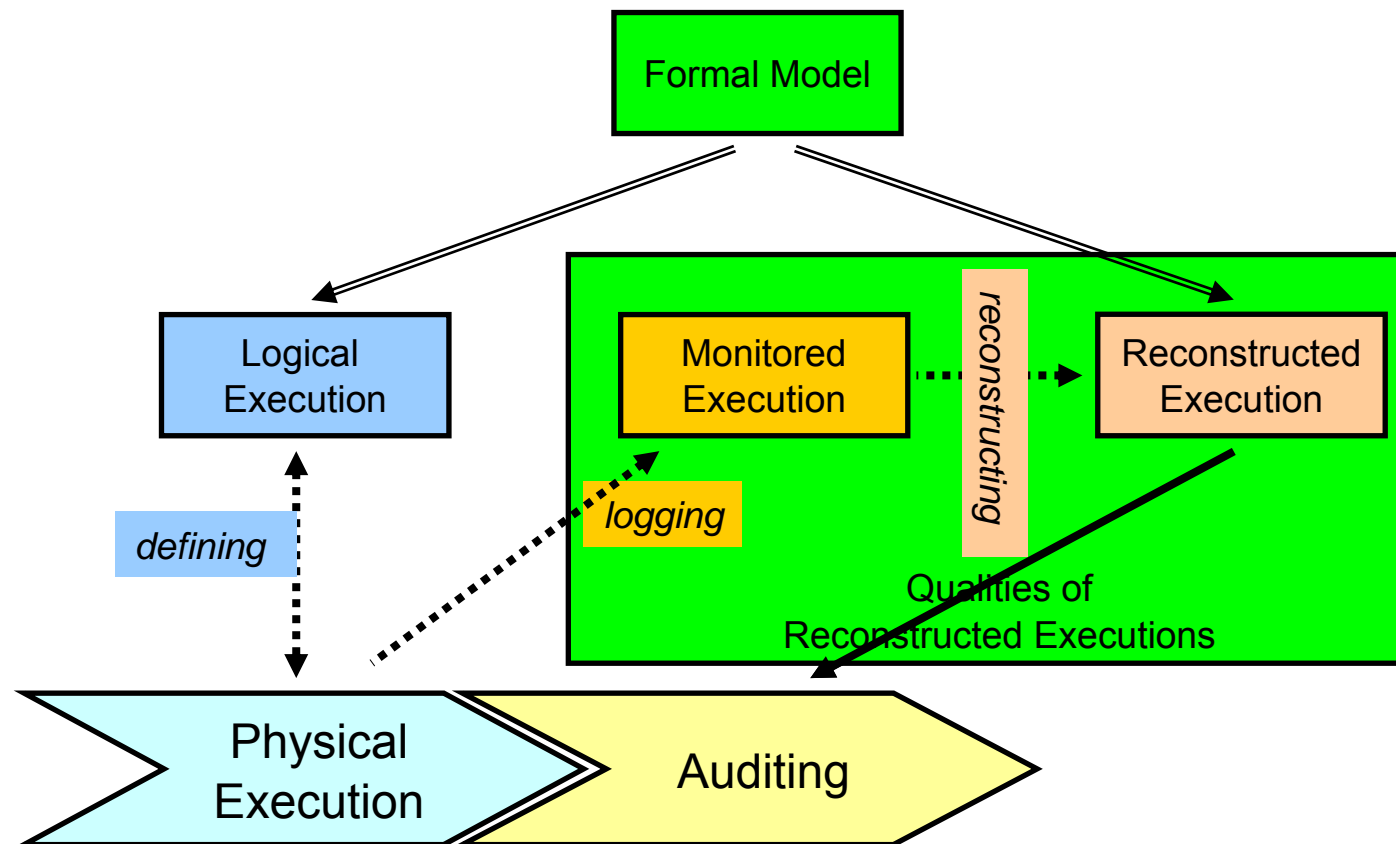
**Can we still audit the processing?**

# **DiALog**

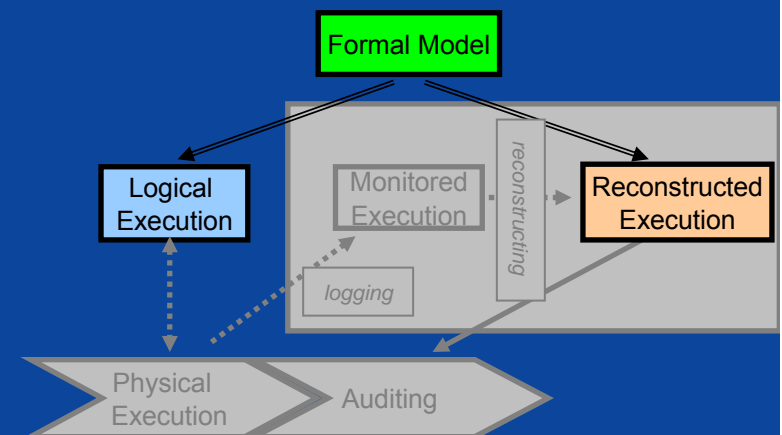
## **Distributed Auditing Logs**

## DiALog

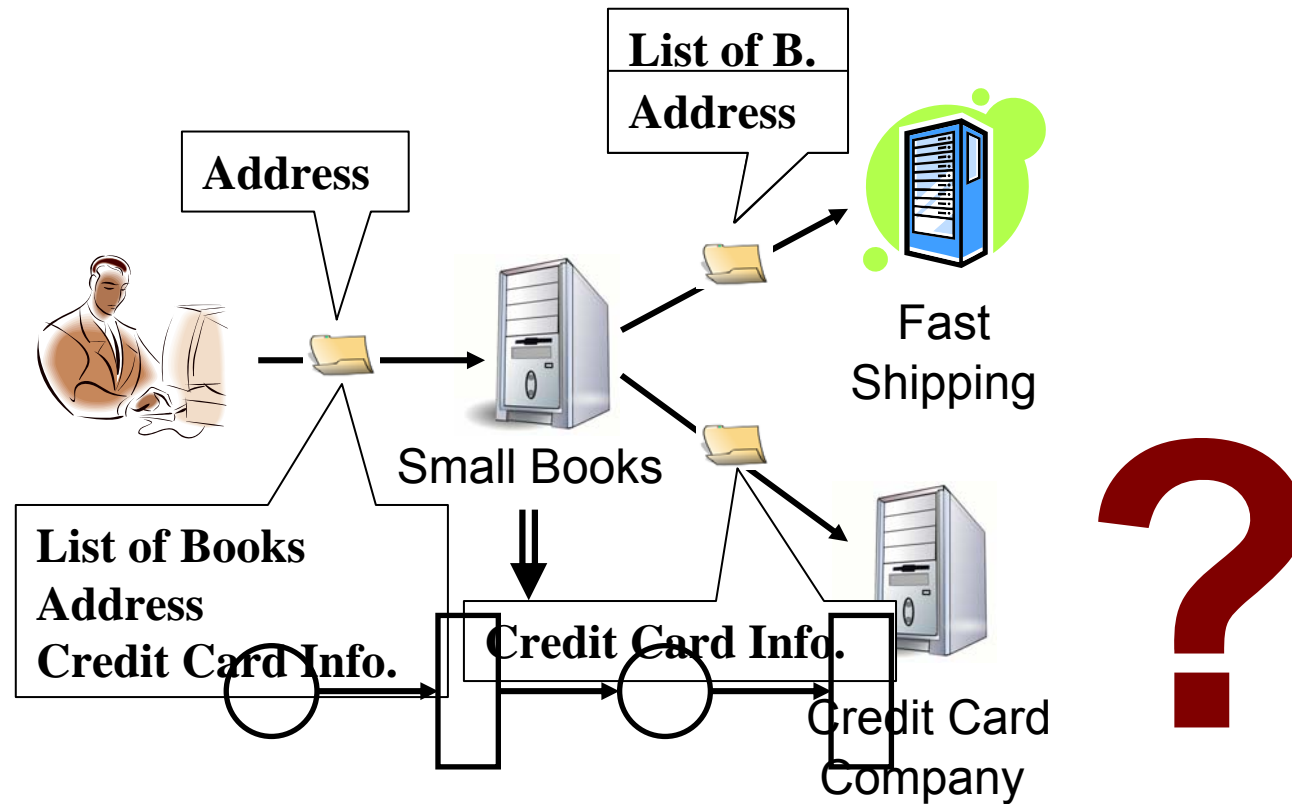
Is a method for auditing the distributed processing of **data items** in distributed systems.



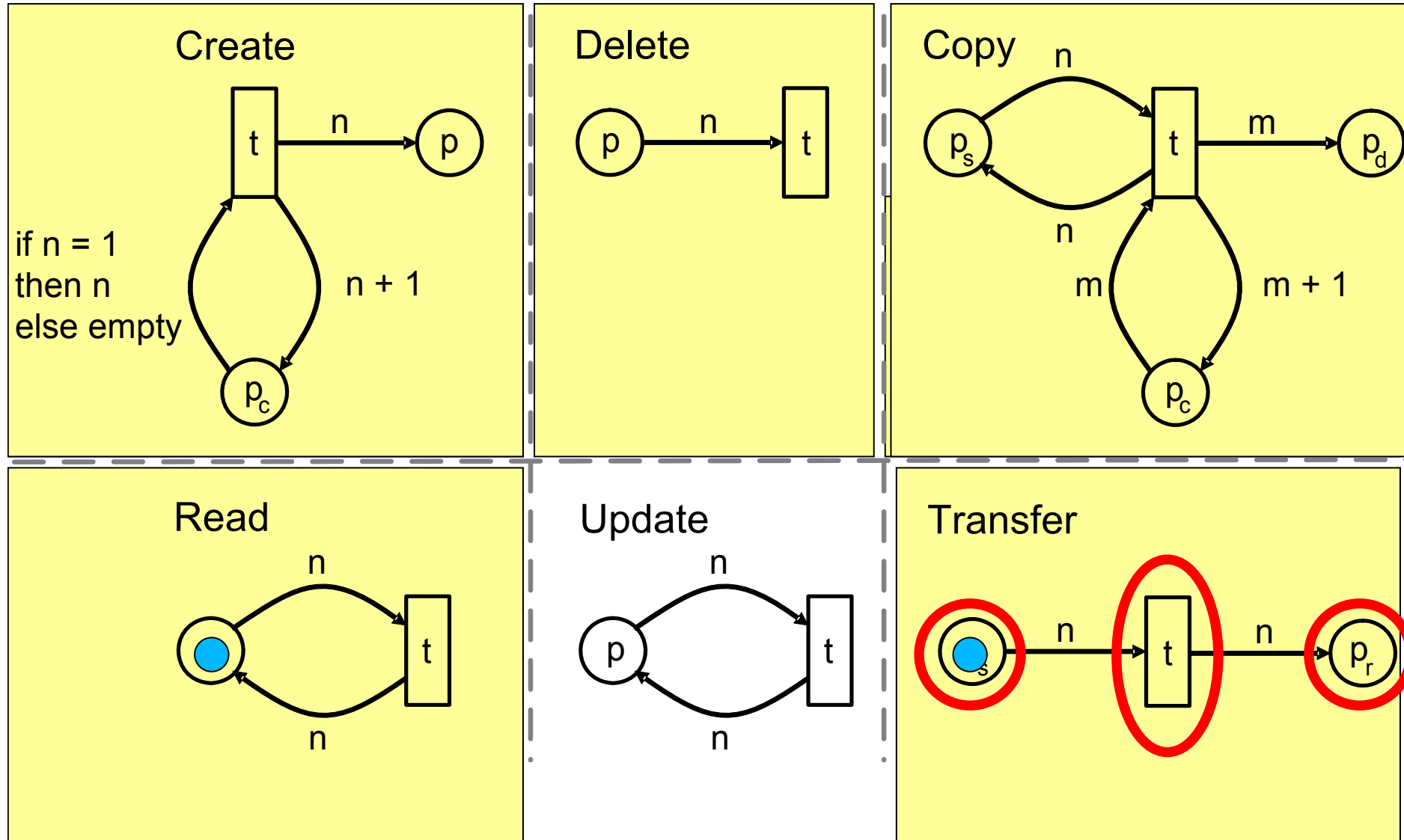
# Formal Model of Distributed Data Processing



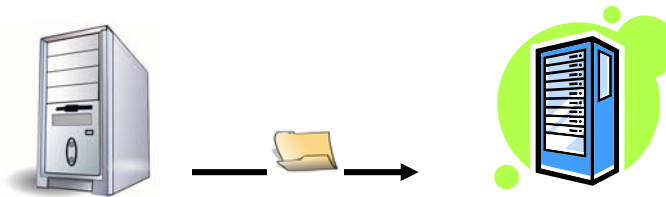
# Book-Store Scenario



**Entities process data by performing actions.**

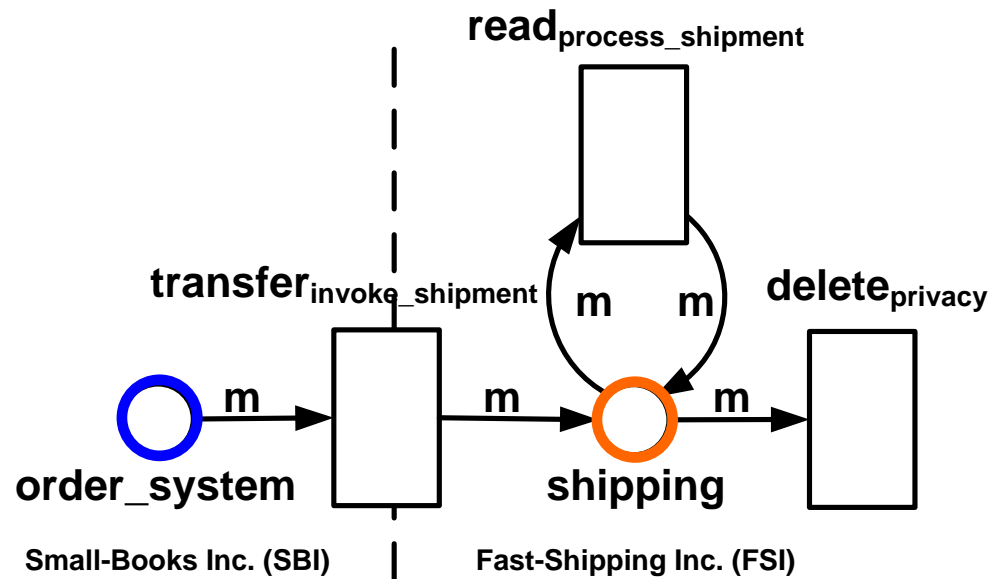
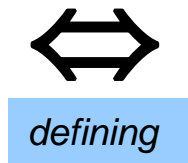


## Logical Execution

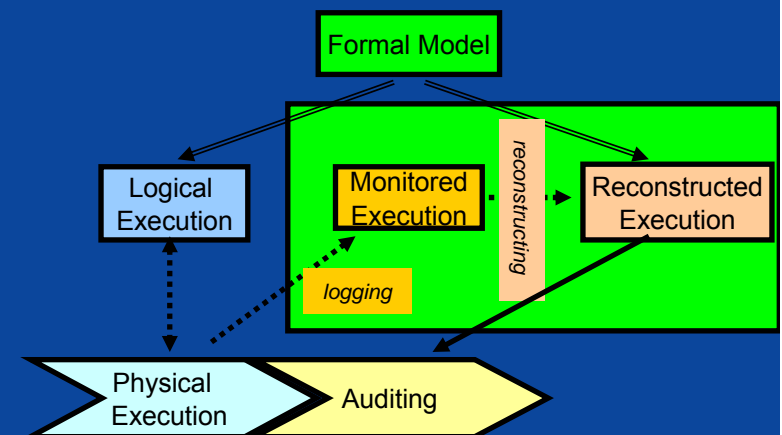


Small Books

Fast Shipping



# Qualities of Reconstructed Executions

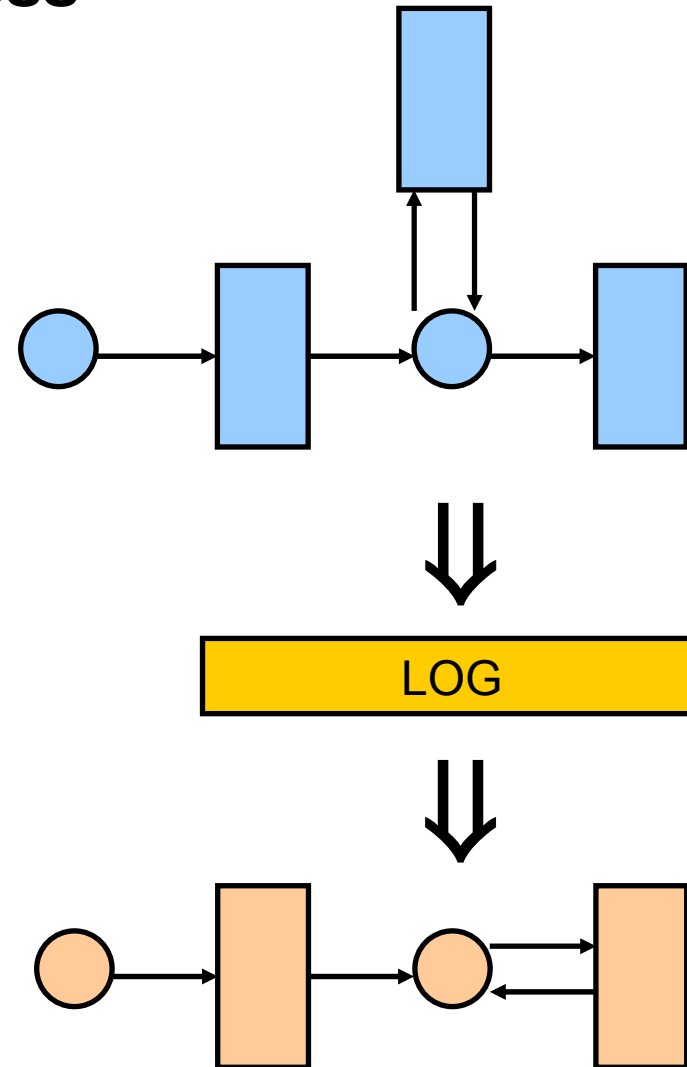


## Soundness

Given:  
a logical execution  $L$

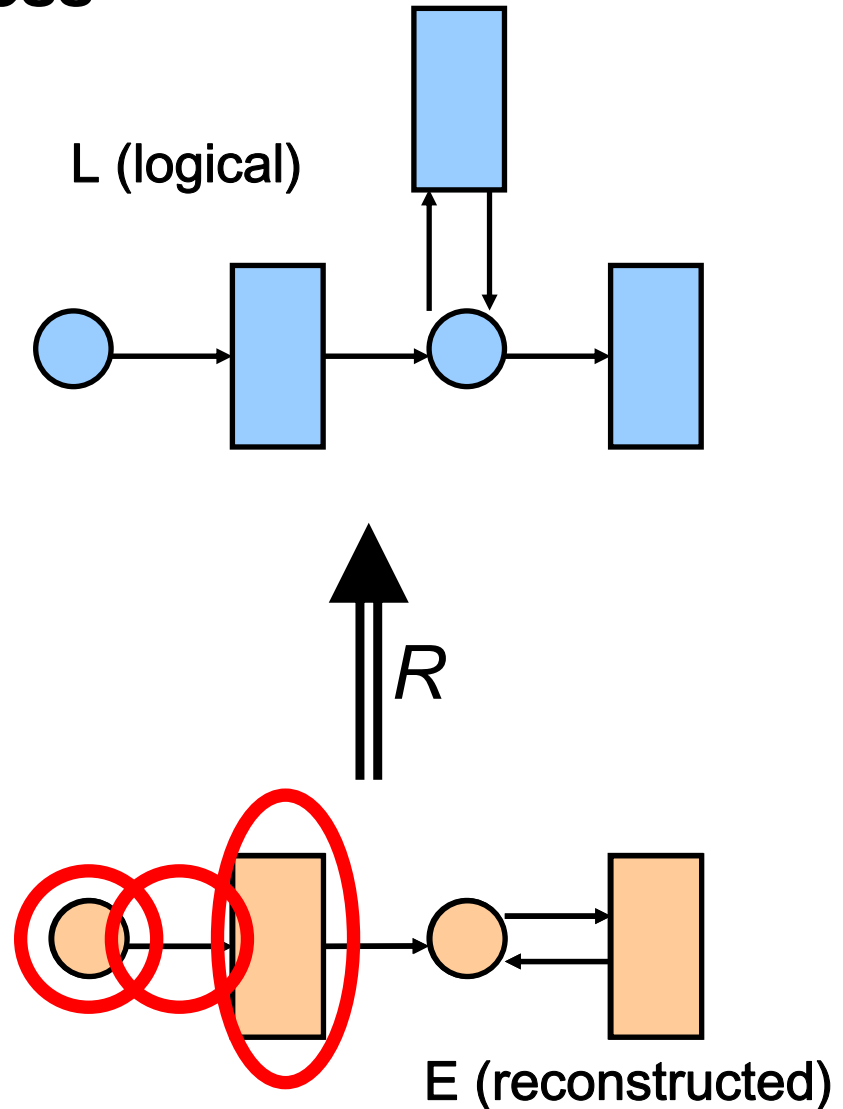
a monitored execution  $M$ , and

a reconstructed execution  $E$



### Soundness

$E$  is sound with respect to  $L$ , if there exists a simulation  $R$  such that for all elements  $e \in E$  there exists an element  $l \in L$  so that  $(e, l) \in R$ .

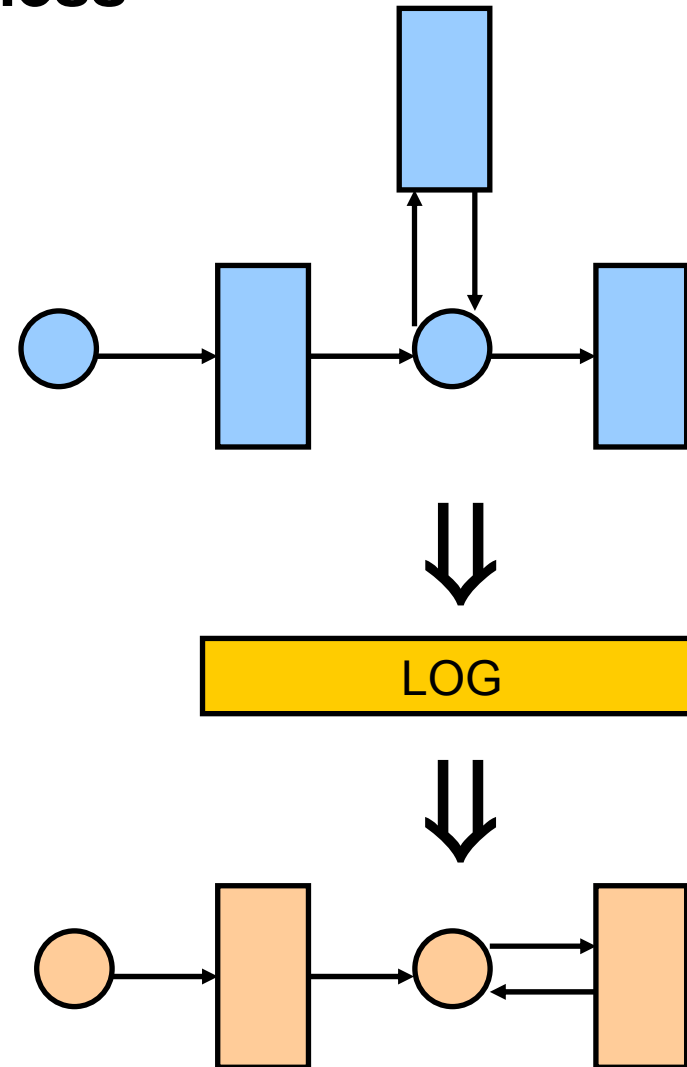


## Completeness

Given:  
a logical execution  $L$

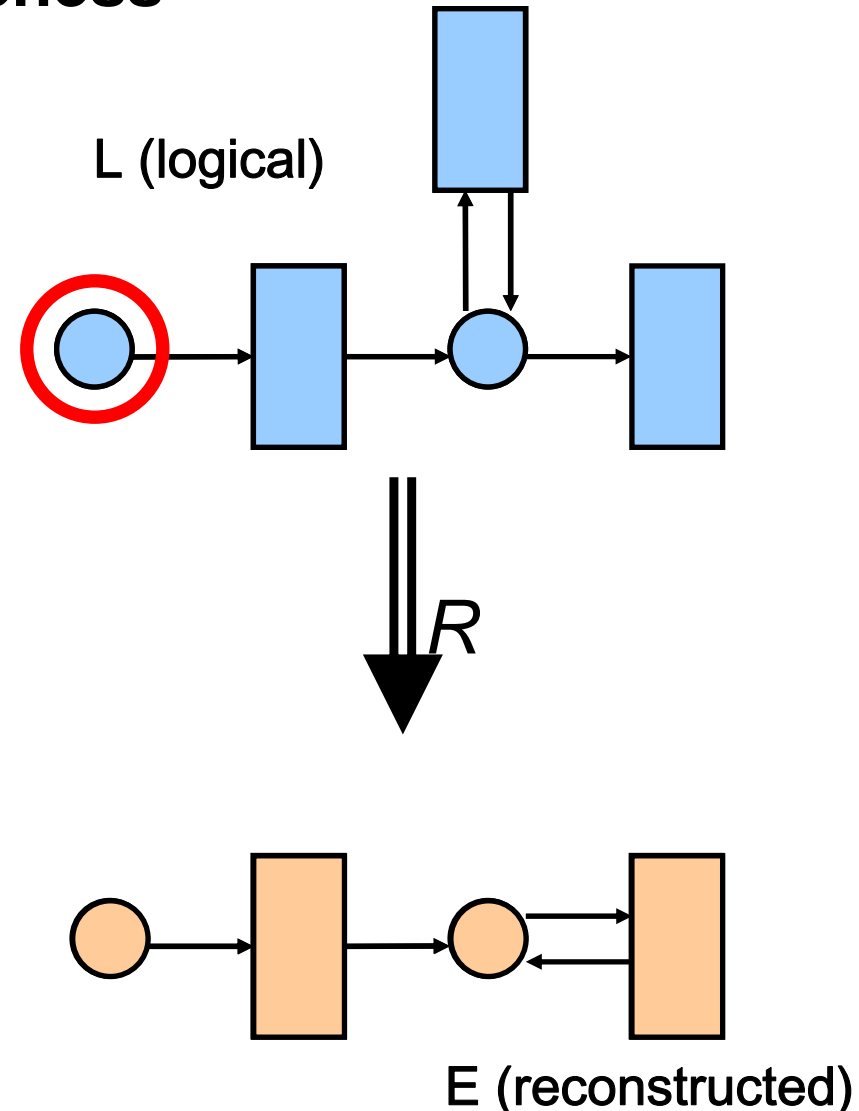
a monitored execution  $M$ , and

a reconstructed execution  $E$



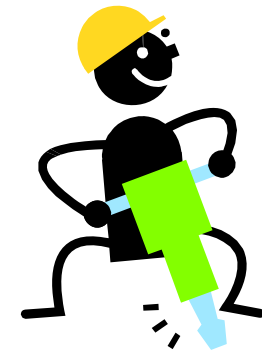
### Completeness

$E$  is complete with respect to the executed subsystem  $S$  of  $L$ , if there exists a simulation  $R$  so that for all elements  $s \in S$  there exists an element  $e \in E$  so that  $(s, e) \in R$ .

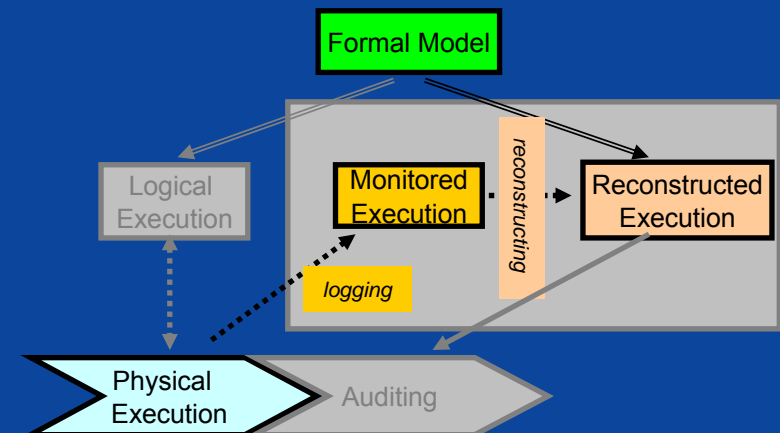


## Requirement for Monitoring Mechanisms

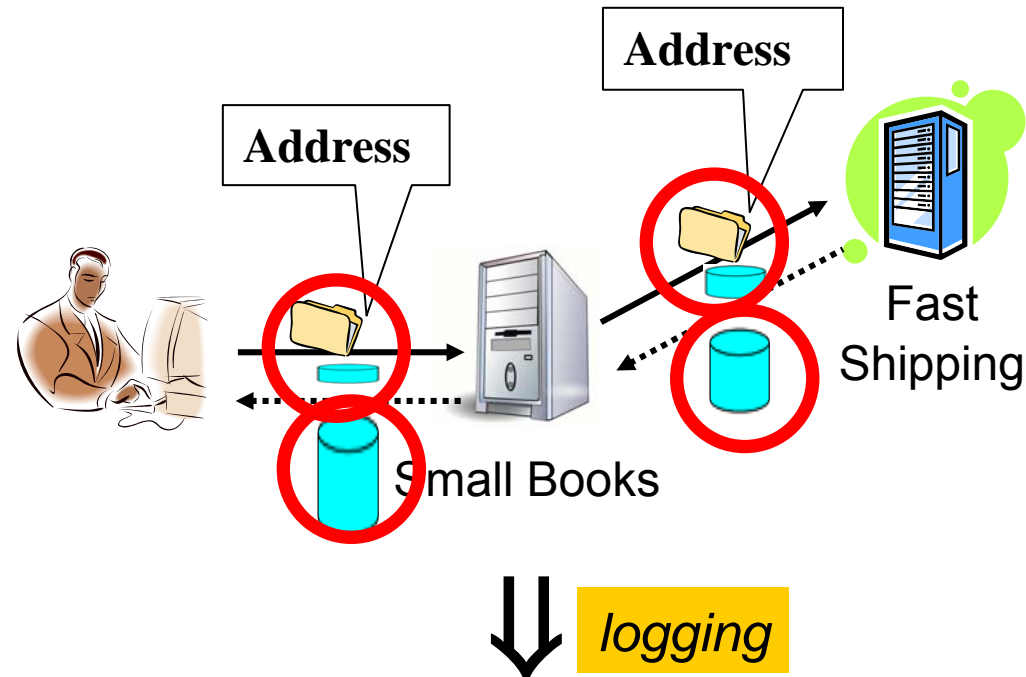
The **monitoring mechanism** must enable the creation of **reconstructed executions**, which are **sound and complete** with respect to the actually executed sub-system of the **logical execution**.



# Monitoring Mechanism Sticky Logging

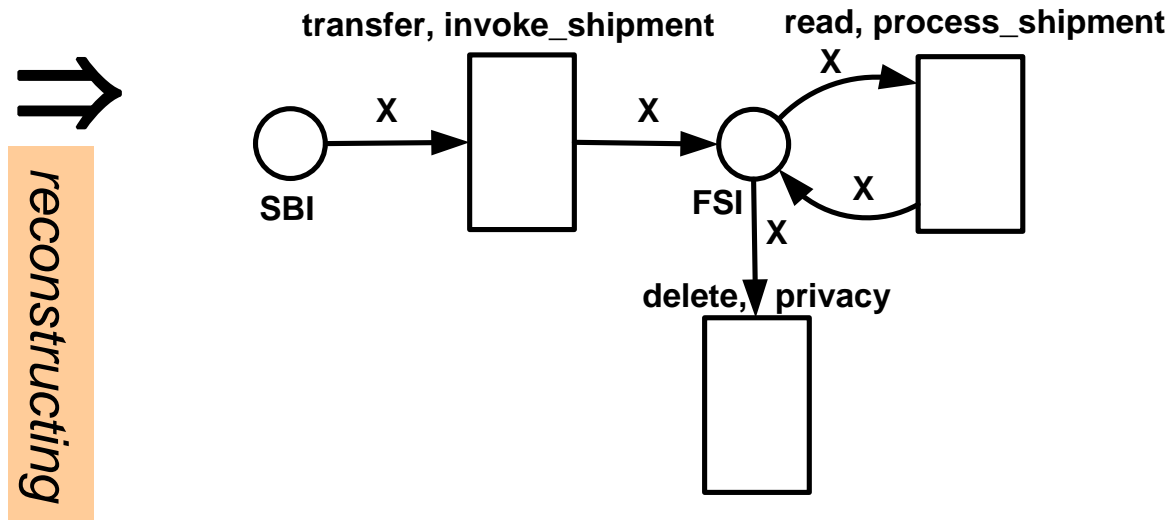
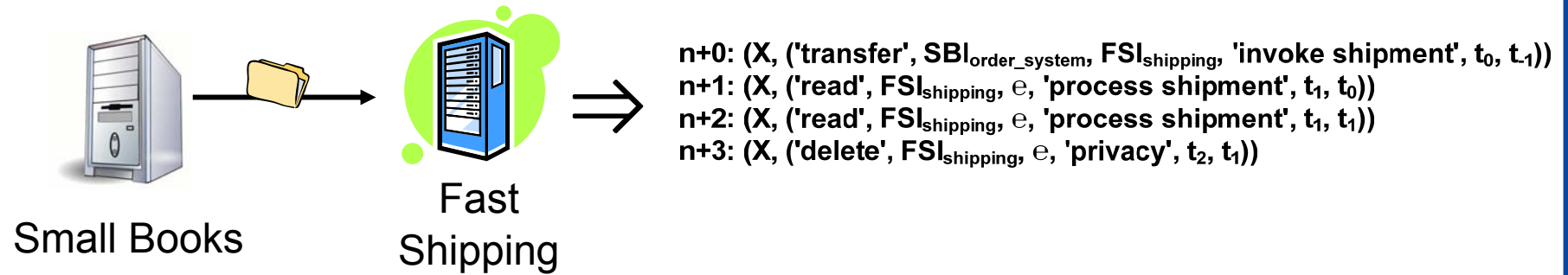


## Monitored Execution

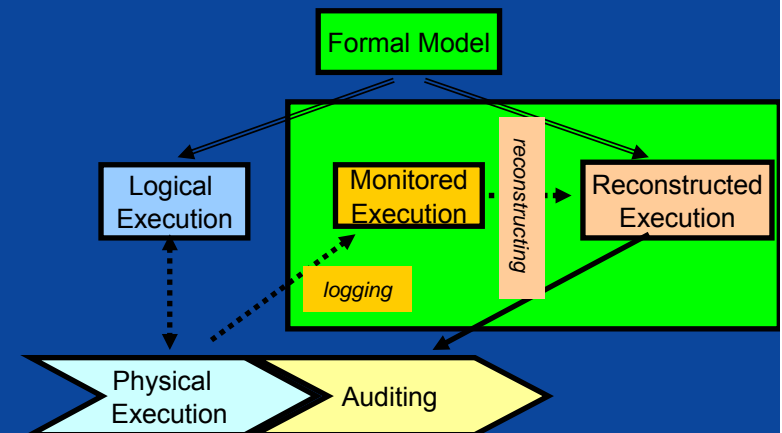


- n+0: (X, ('transfer',  $SBI_{order\_system}$ ,  $FSI_{shipping}$ , 'invoke shipment',  $t_0$ ,  $t_{-1}$ ))**
- n+1: (X, ('read',  $FSI_{shipping}$ , e, 'process shipment',  $t_1$ ,  $t_0$ ))**
- n+2: (X, ('read',  $FSI_{shipping}$ , e, 'process shipment',  $t_1$ ,  $t_1$ ))**
- n+3: (X, ('delete',  $FSI_{shipping}$ , e, 'privacy',  $t_2$ ,  $t_1$ ))**

## Reconstructed Execution



# Proof of Soundness and Completeness



## Proposition

A reconstructed execution created by means of the sticky logging mechanism is sound and complete regarding the executed subsystem of the logical execution.

## Proof Sketch

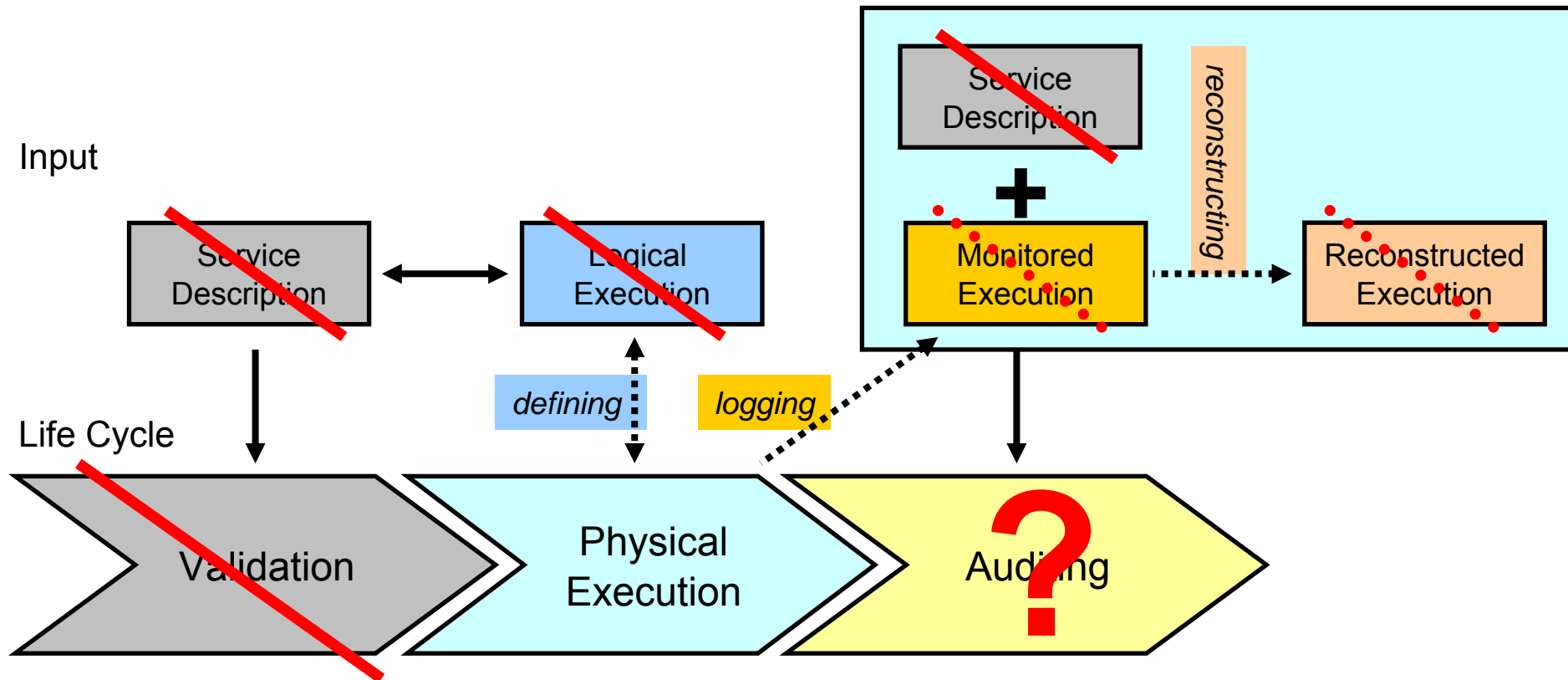
**Induction:** over the structural length of the CPN.

**Induction basis:** a logical execution consisting only of a create action.

**Induction step:** extending a given logical execution, which fulfills the proposition, by one action (for each category).

# Conclusion

# Life Cycle of Controlling Distributed Data Processing



**Can we still audit the processing?**

## Conclusion

***Yes, the auditing can be done  
by means of DiALog and Sticky Logging!***



**Thank You for Your Attention!**

<http://isweb.uni-koblenz.de/StickyLogging>

Supported by X-Media (IST, FP6-26978) and Neon (IST, FP6-27595).