Component Integration using Composition Contracts with Exception Handling

Ricardo de Mendonça da Silva
Paulo Asterio de C. Guerra (*presenter*)
and Cecília M.F. Rubira
Institute of Computing – University of Campinas – Brazil

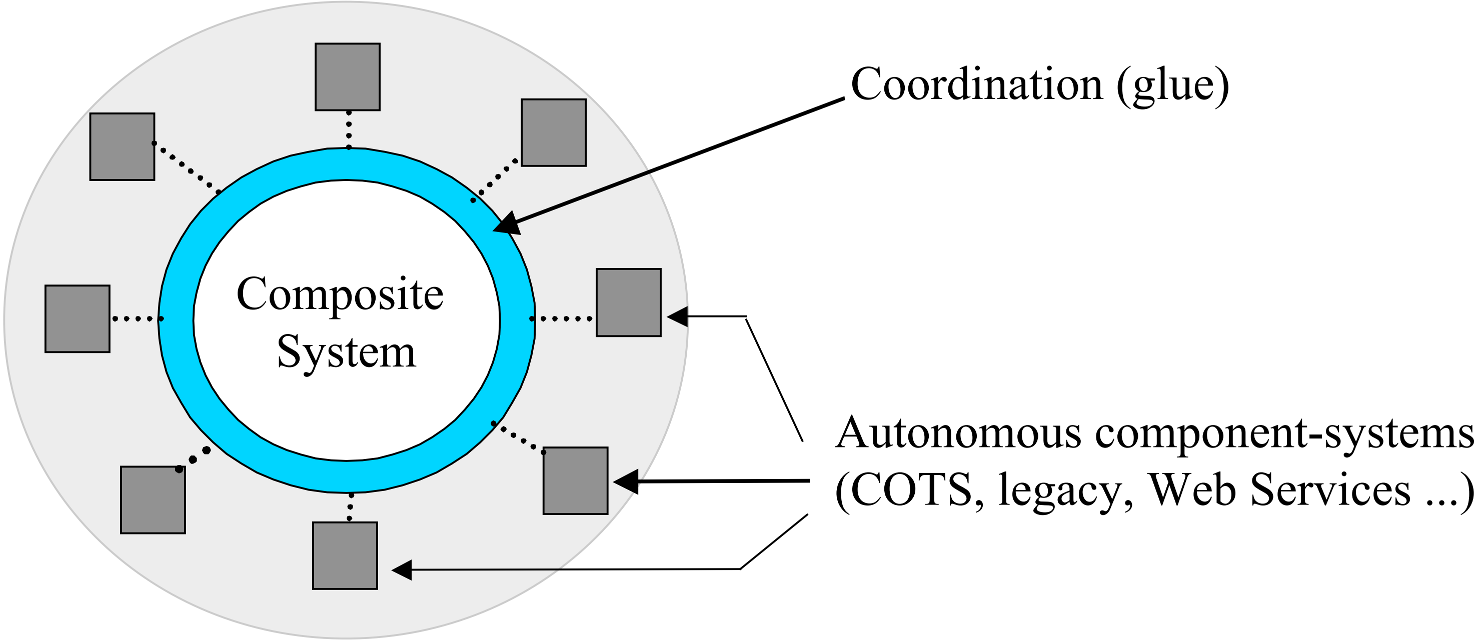
ECOOP'2003 – Workshop 01
Exception Handling in Object Oriented Systems EHOOS'2003
Darmstadt – Germany – July, 21th 2003

Summary

- Introduction
- Background
- The Proposed Architecture
- Case Study
- Conclusions and Related Work

Introduction: Motivation

To build adaptable and dependable systems based on composition of autonomous component-systems.

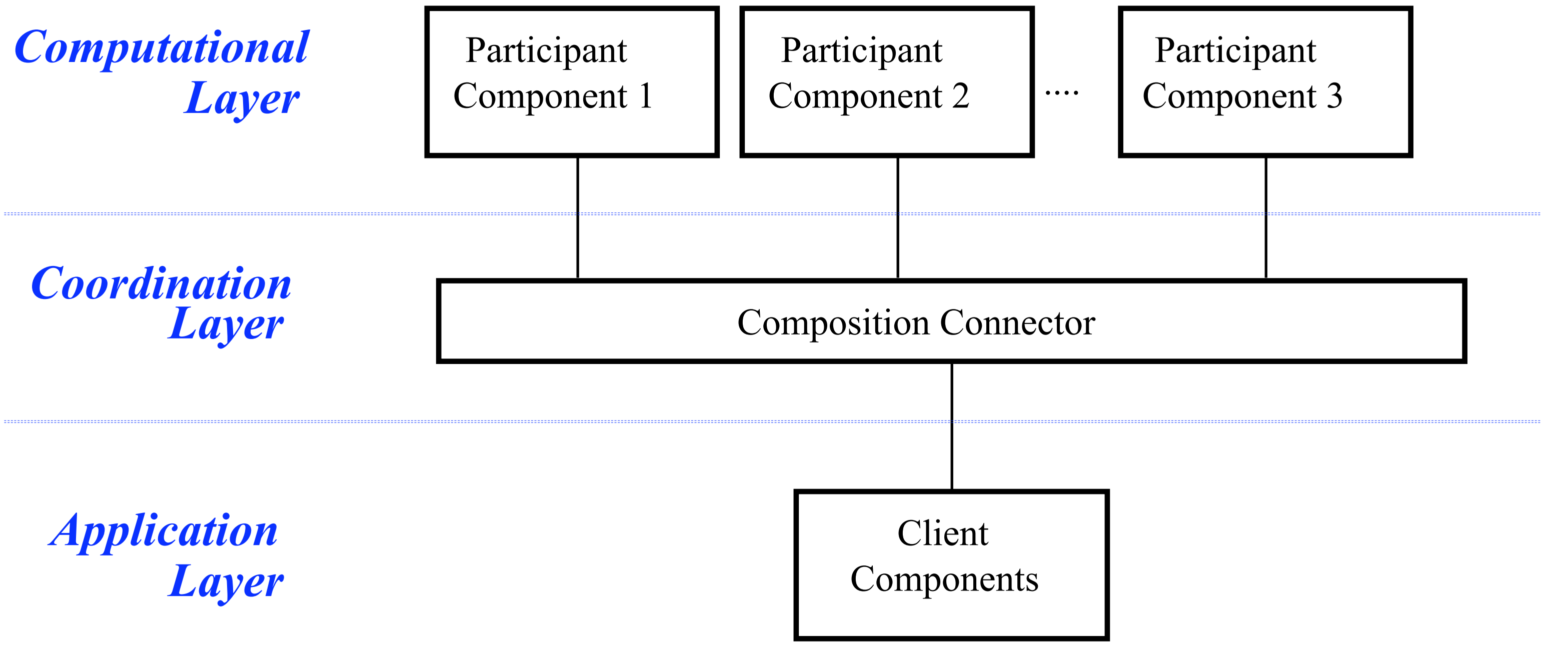


Composite System

Coordination (glue)

Autonomous component-systems (COTS, legacy, Web Services ...)

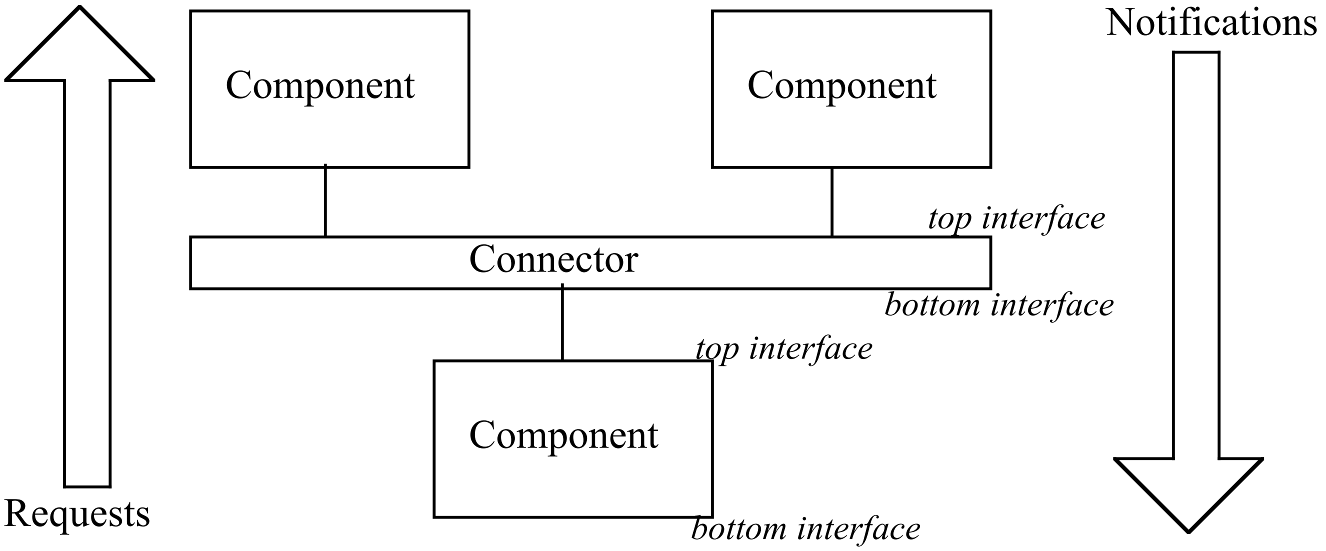
Introduction: Overall Solution



- C2 Architectural Style
- Coordination Contracts (Composition)
- CA Action (Exception Handling)

Background: C2 Style

- Components unaware of each other
- Asynchronous messages mediated by connectors



Background: Composition Contracts

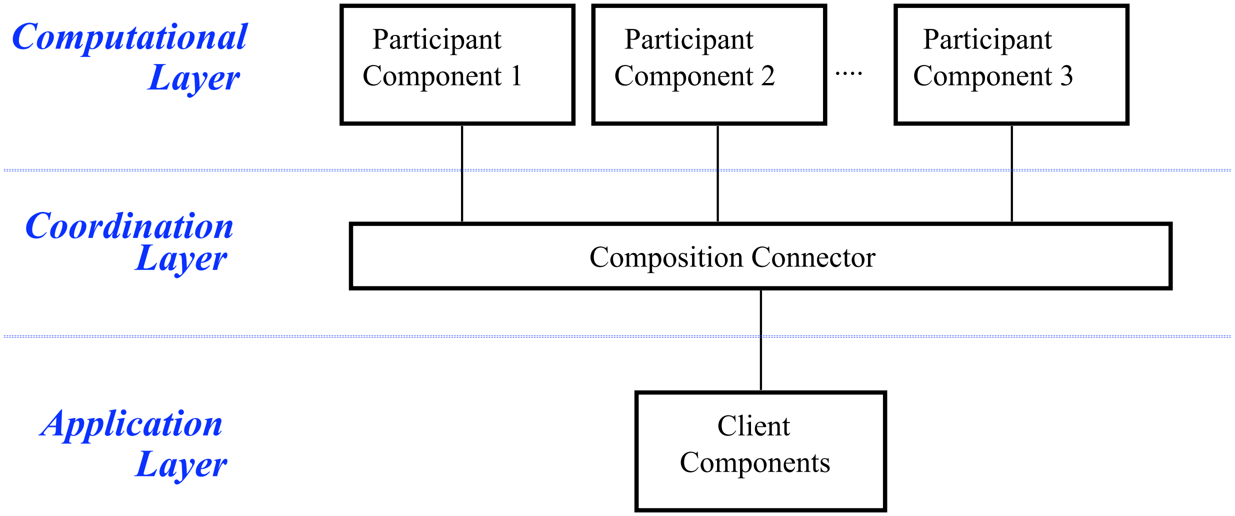
- A connection between objects through which rules and constraints are imposed on joint behavior
- Consist of a prescription of *coordination effects* that will be imposed on a collection of *partners* when the occurrence of one of the contract *triggers* is detected in the system

Background: CA Action

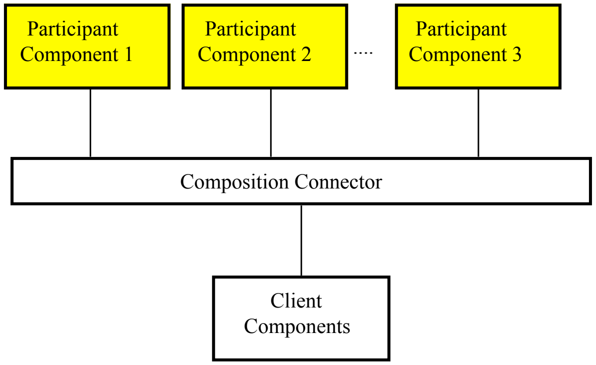
- Coordinates participants
- Exception resolution scheme
- Cooperative error recovery
- Forward and backward recovery
- Normal, exceptional, abort, failure results
- Action nesting

The Proposed Architecture

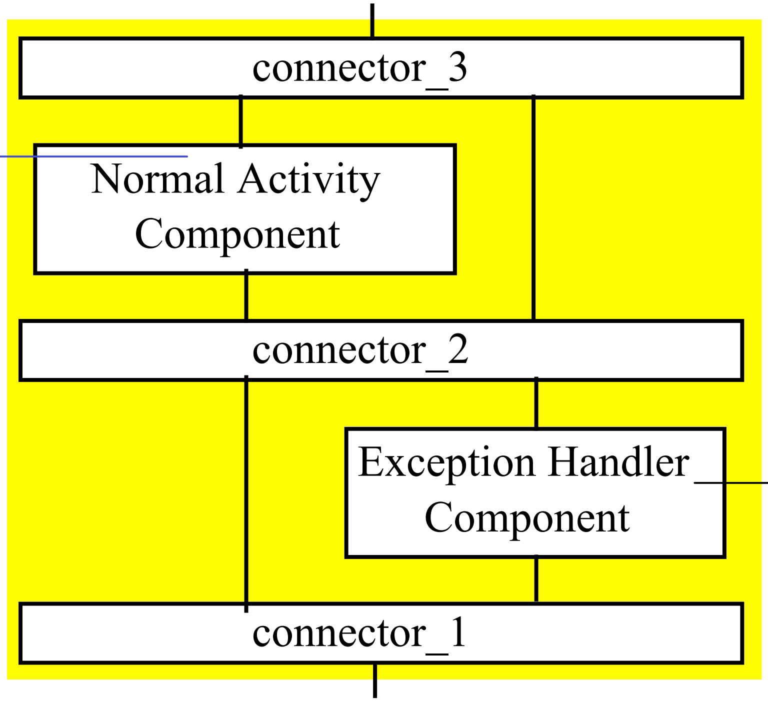
- The Participan Components
- The Composition Connector
- Message Flows (Normal & Exceptional)



The Participant Components

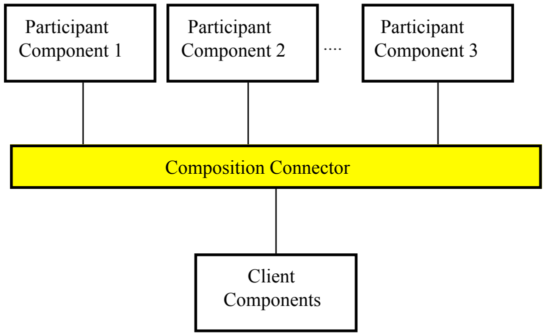


Wrappers Component-System



Local Recovery & Compensation Actions

The Composition Connector



Participant Component 1

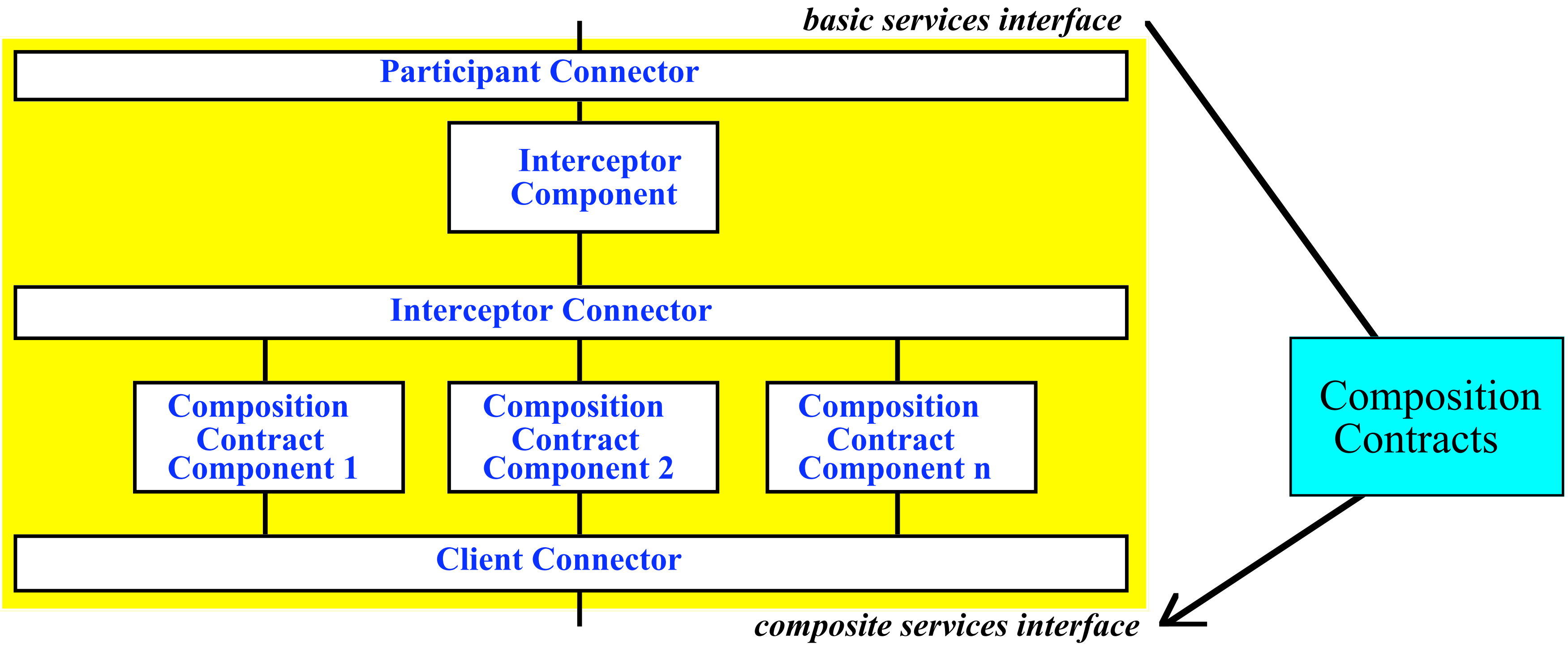
Participant Component 2

····

Participant Component 3

Composition Connector

Client Components



basic services interface

Participant Connector

Interceptor Component

Interceptor Connector

Composition Contract Component 1

Composition Contract Component 2

Composition Contract Component n

Client Connector

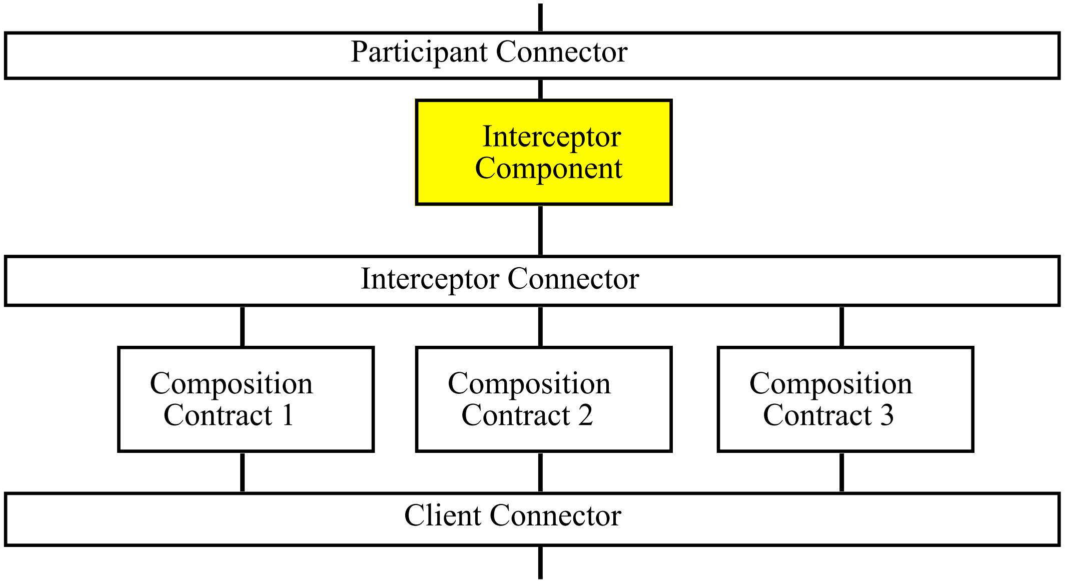
Composition Contracts

composite services interface

Composition Contract Definition

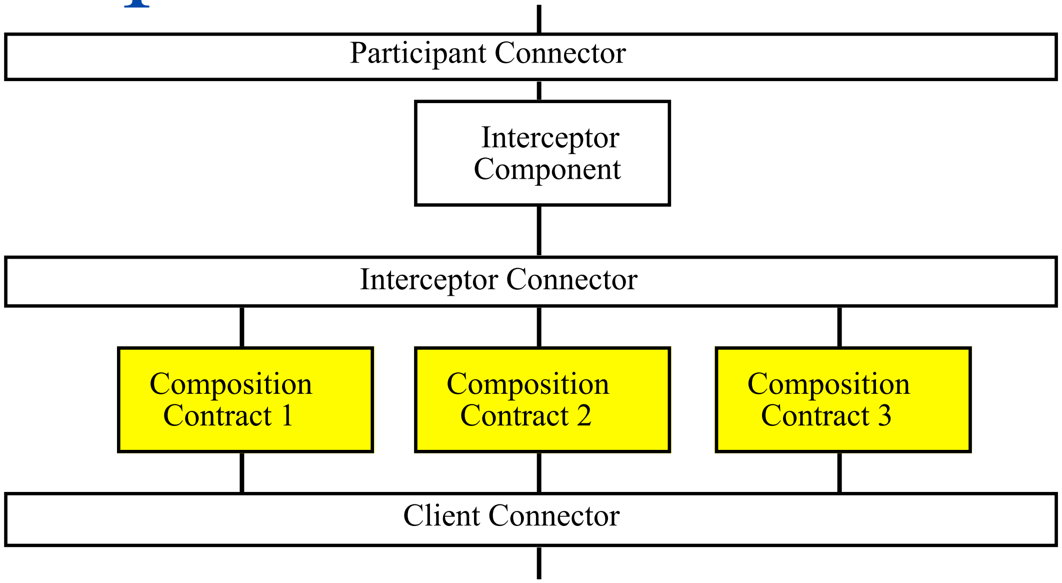
- Contract Triggers
- Service Composition
- Exceptional Conditions
- Error Recovery (Exceptional Contracts)

The Interceptor Component



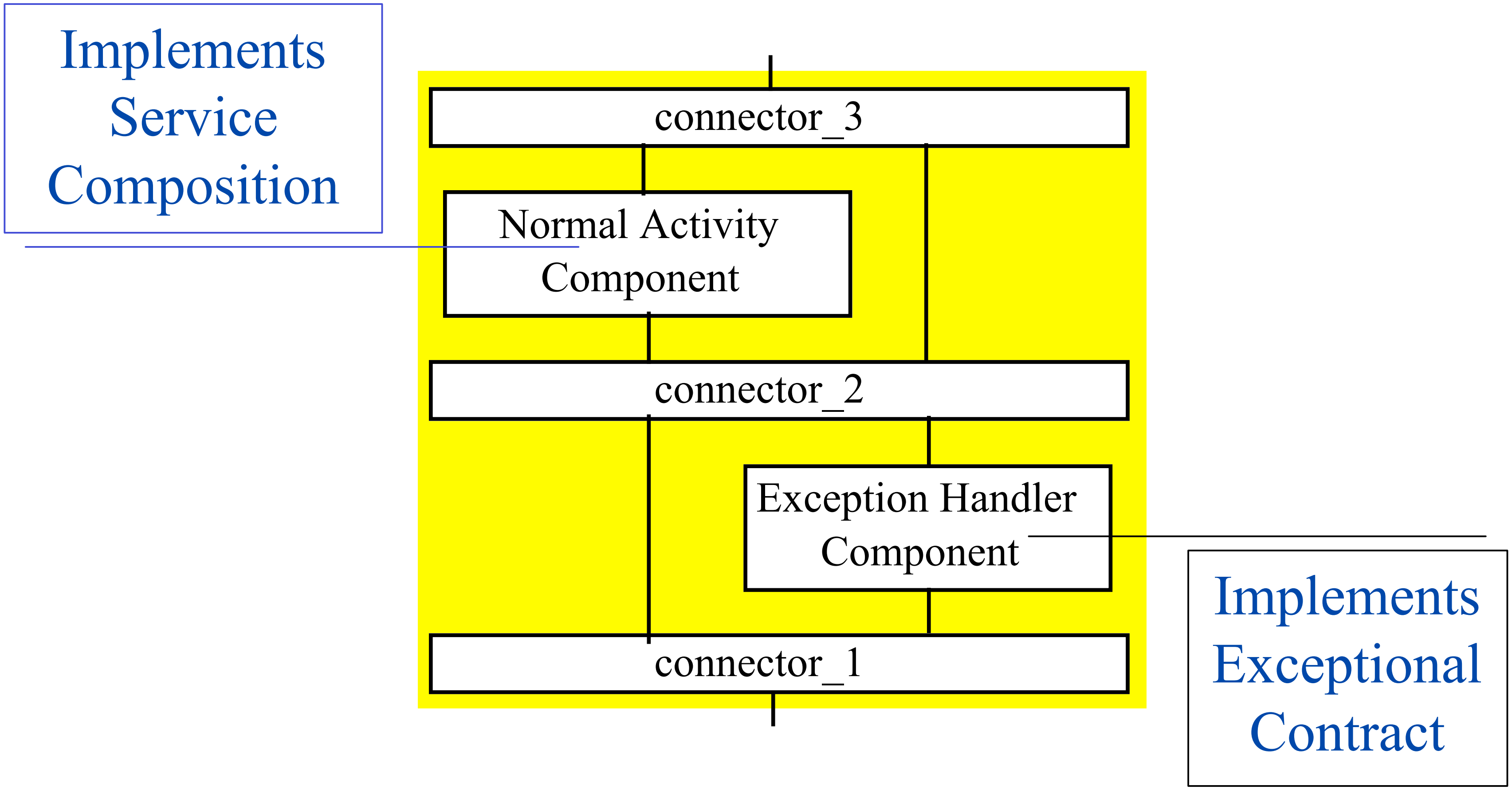
- Intercepts messages to / from participants
- Monitors contract triggers (service requests)
- Notifies a composition contract when a trigger is enabled

The Composition Contract Component

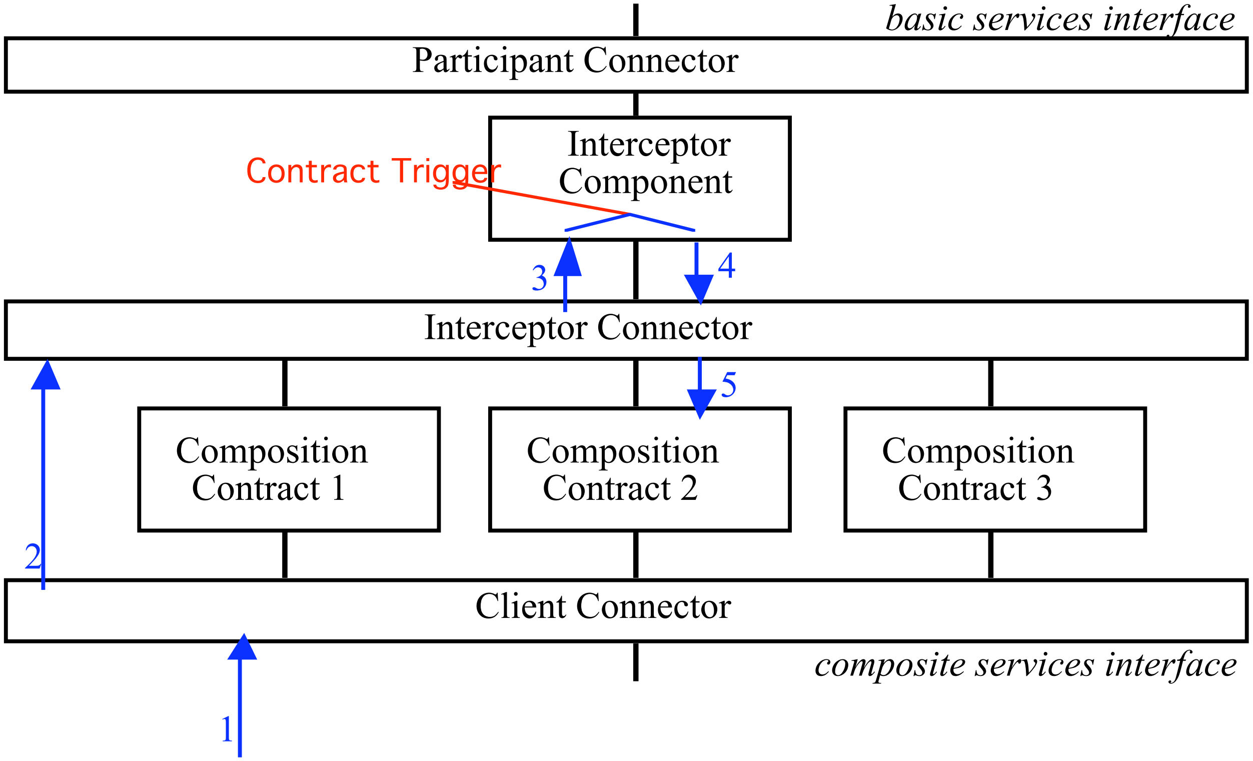


- Implements a composite service as a relaxed CA Action (non-transactional external objects)
- Resolves concurrent exceptions and coordinates error recovery

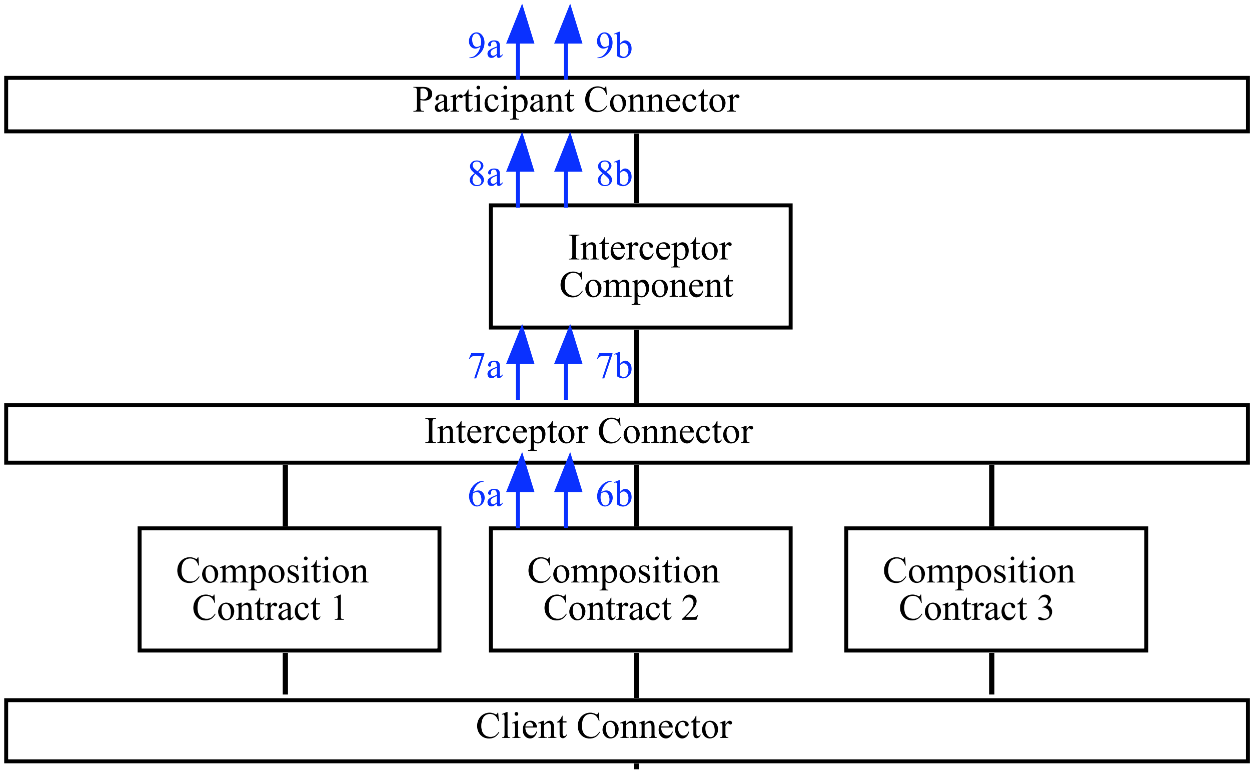
The Composition Contract Component



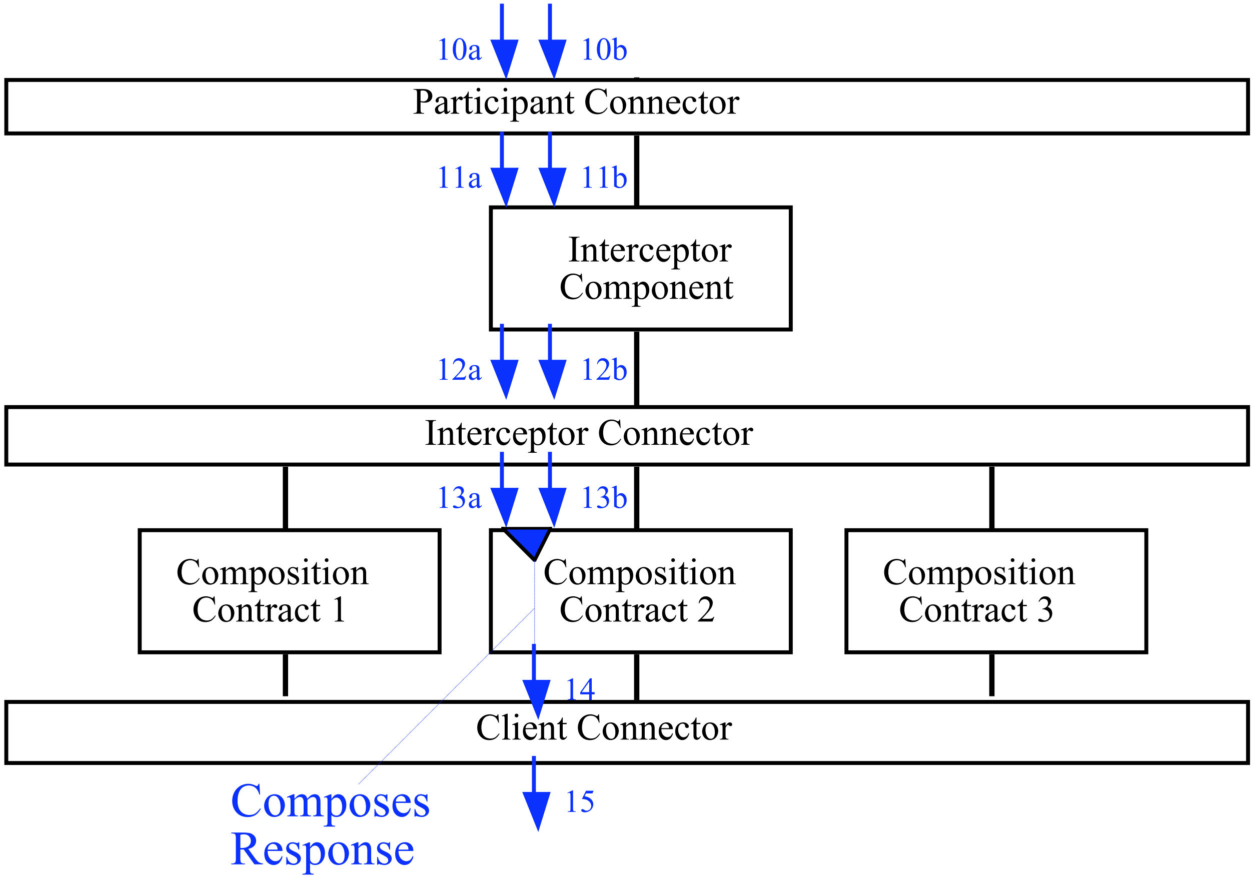
Intercepting a Service Request



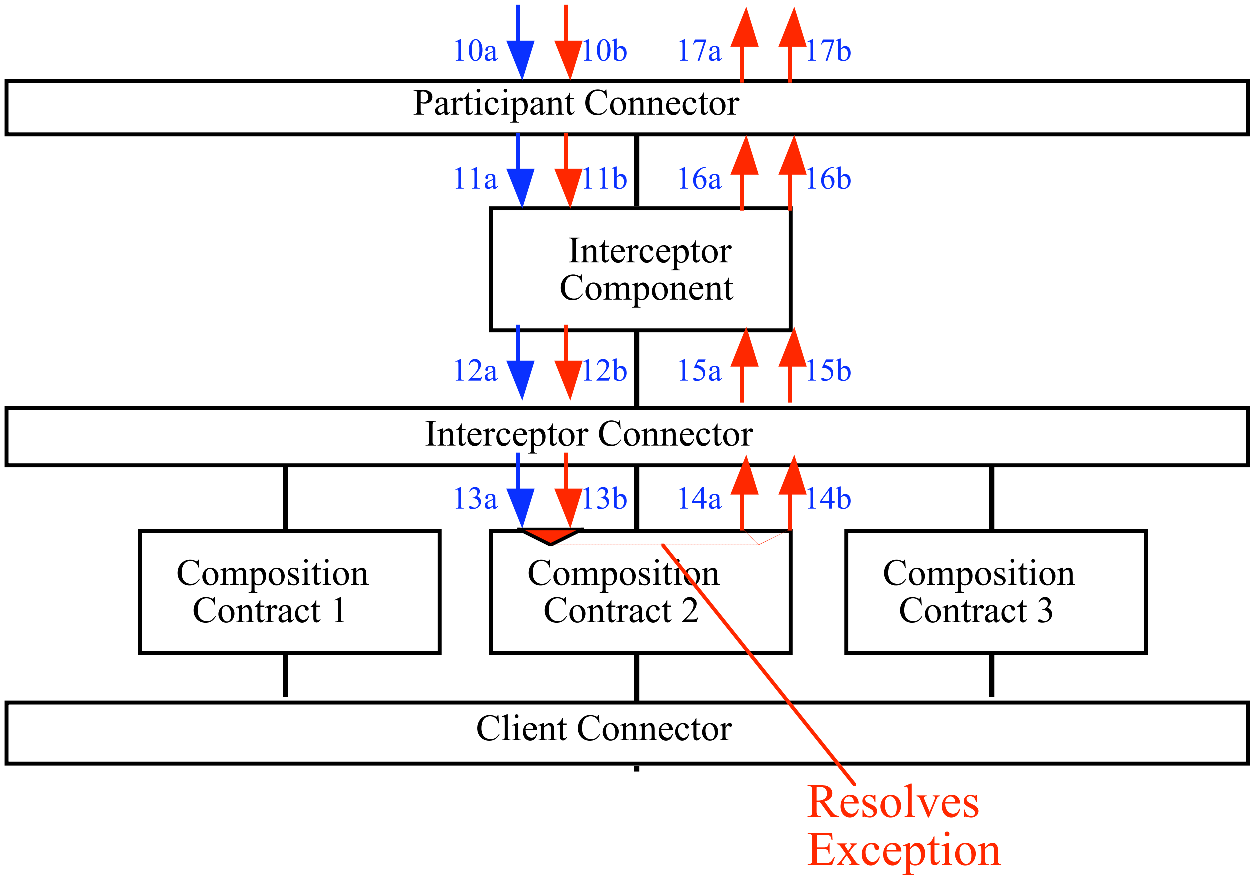
Service Composition: Outgoing Requests



Collecting Responses Normal Flow

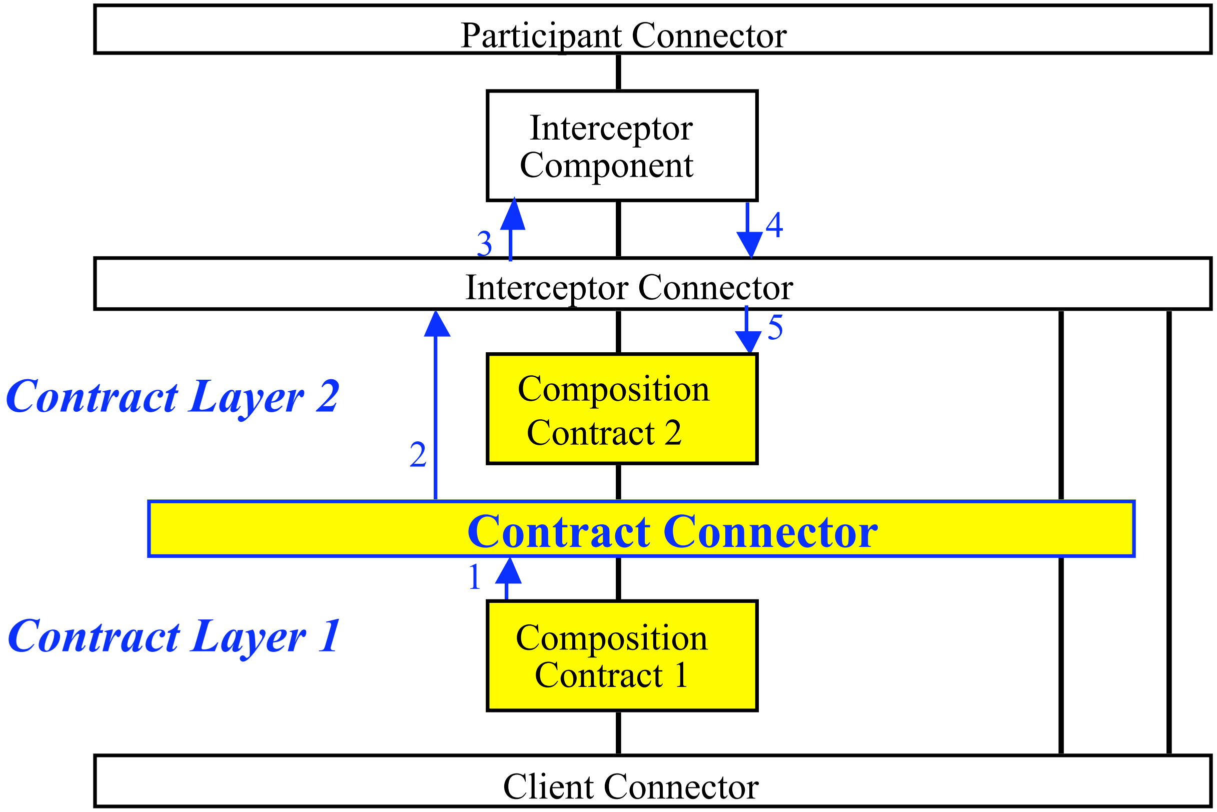


Exceptional Flow
Cooperative Error Recovery

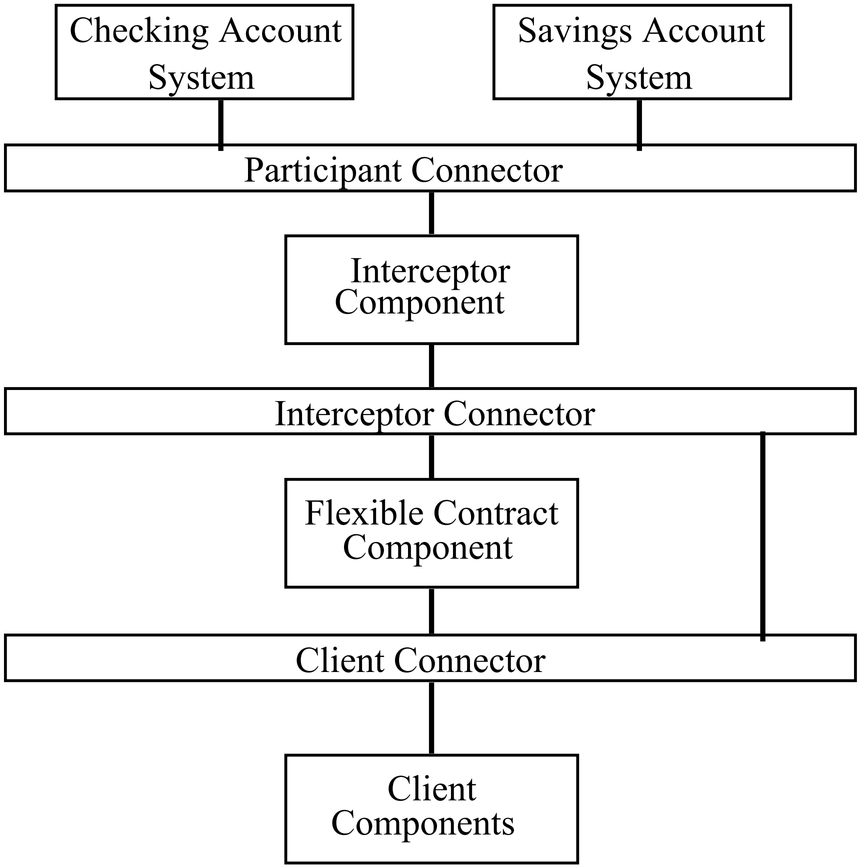


Nested Composition

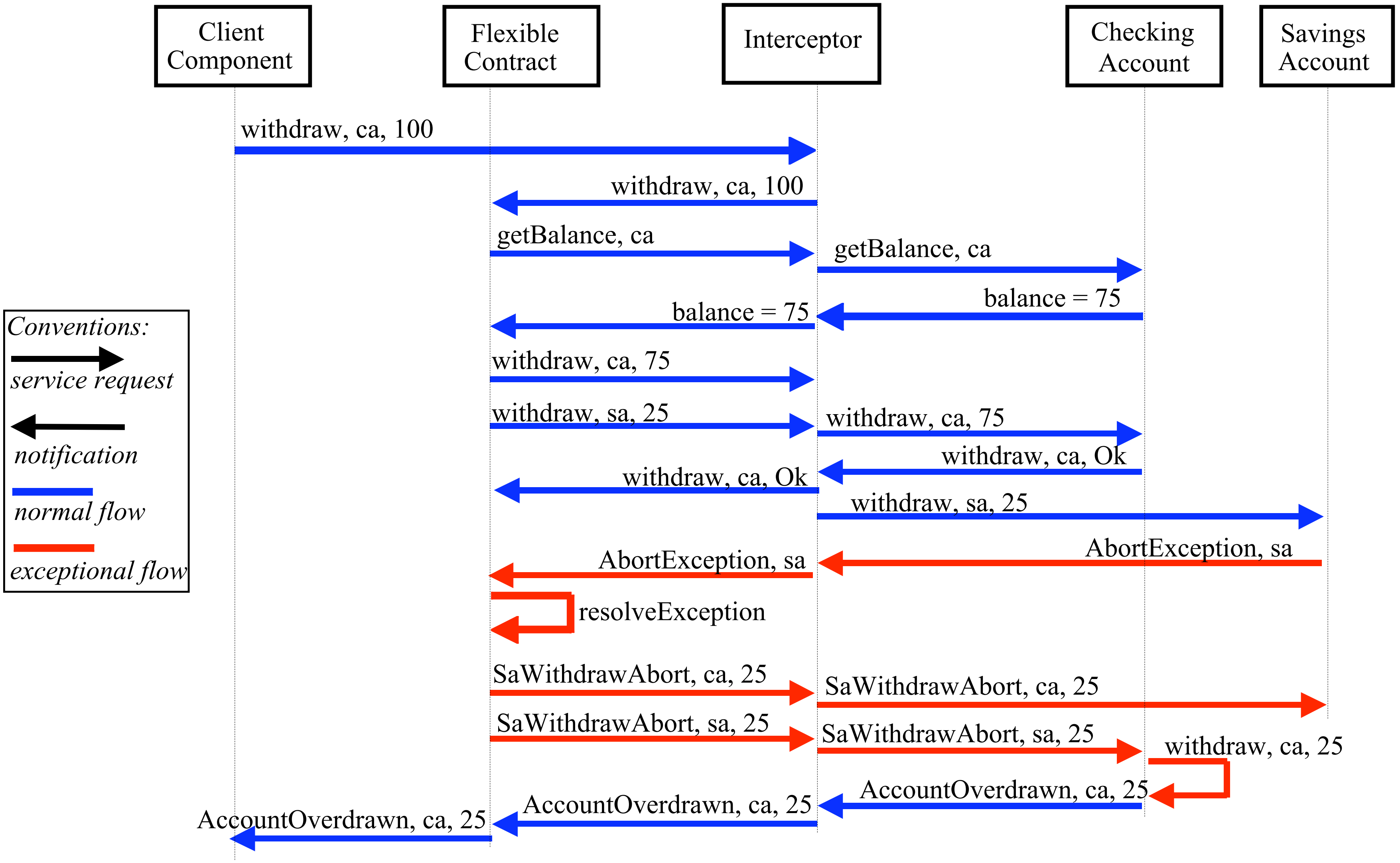
A service request sent by a composition contract activates another composition contract.



Case Study



Flexibel Contract:
Avoids overdrafts of a checking account using funds from an associated savings account.



Client Component

Flexible Contract

Interceptor

Checking Account

Savings Account

withdraw, ca, 100

withdraw, ca, 100

getBalance, ca

getBalance, ca

balance = 75

balance = 75

withdraw, ca, 75

withdraw, sa, 25

withdraw, ca, 75

withdraw, ca, Ok

withdraw, ca, Ok

withdraw, sa, 25

AbortException, sa

AbortException, sa

resolveException

SaWithdrawAbort, ca, 25

SaWithdrawAbort, ca, 25

SaWithdrawAbort, sa, 25

SaWithdrawAbort, sa, 25

withdraw, ca, 25

AccountOverdrawn, ca, 25

AccountOverdrawn, ca, 25

AccountOverdrawn, ca, 25

Conventions:

→ *service request*

← *notification*

🟦 *normal flow*

🟥 *exceptional flow*

Conclusions and Related Work

- Architectural solution for dependable software systems out of concurrent autonomous component-systems
- Adaptability, extensibility and reliability
- Adapted CA Action to service oriented approach to system’s software architecture
- WSCA – CA Action for WEB Services composition (no coordination contract, not restricted to WEB)
- Pires proposes a layered architecture for sequential composition with backward error recovery