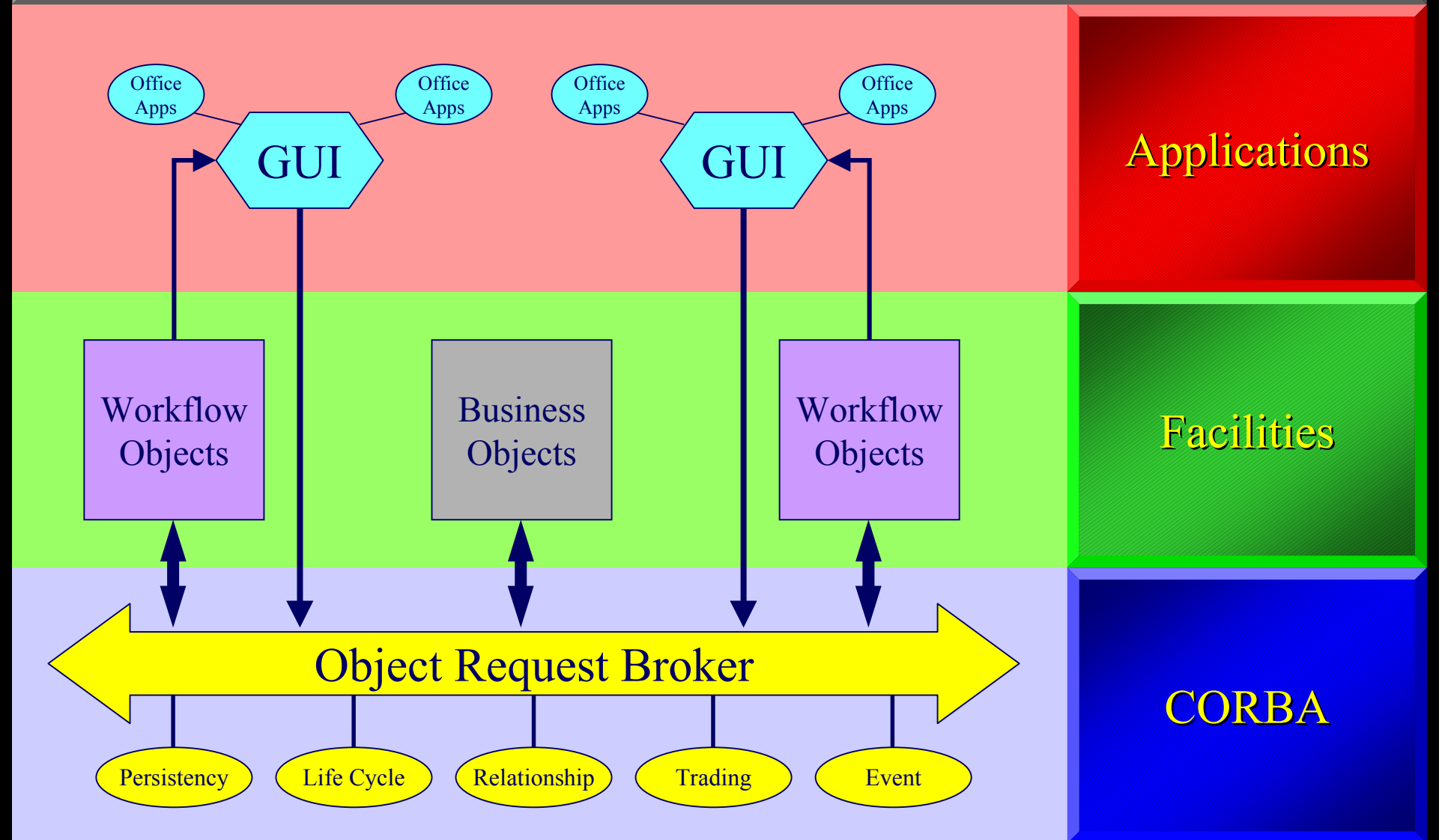


WASA₂ Demonstration

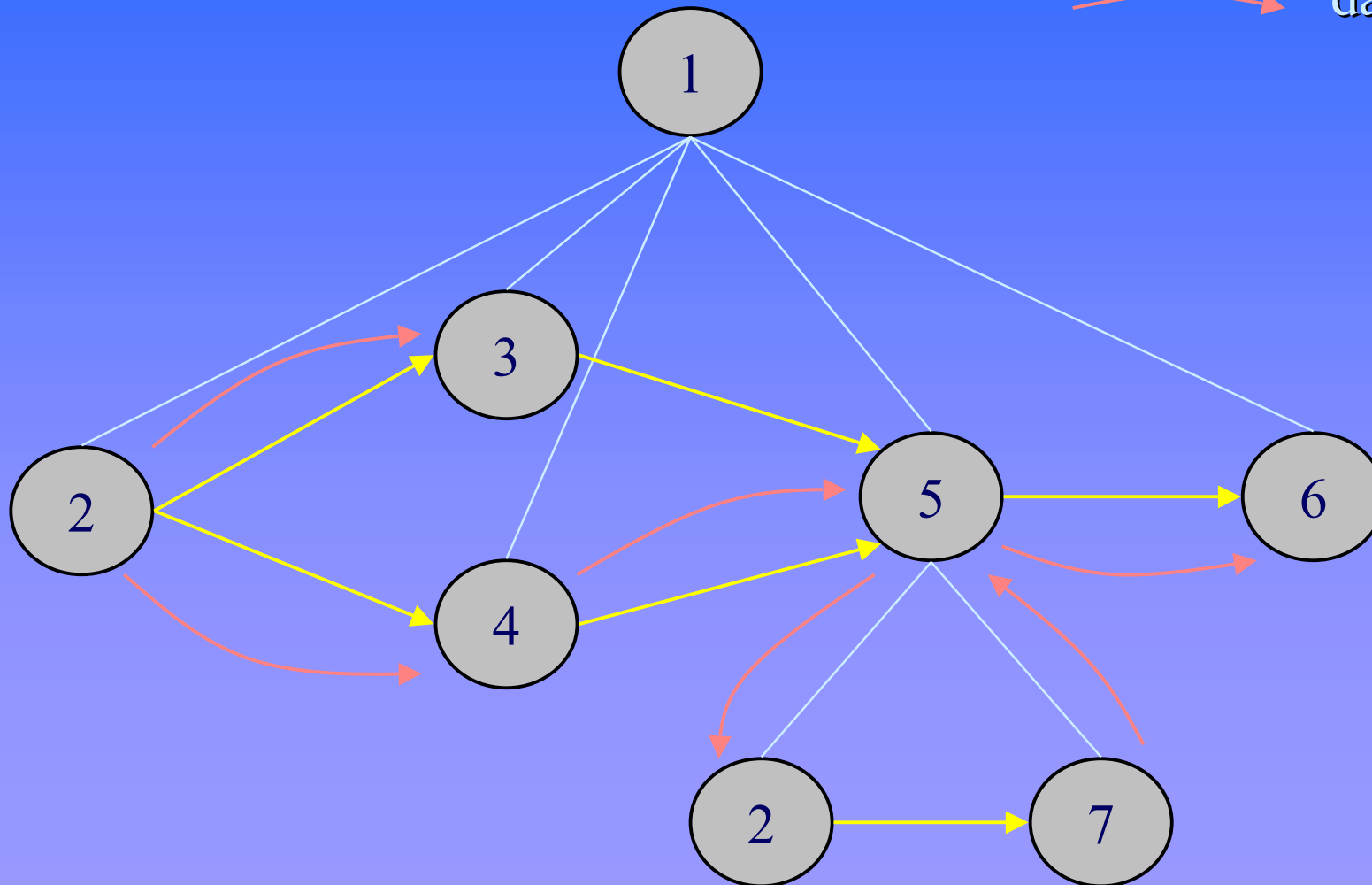
G. Vossen, M. Weske
SIGMOD '99

WASA₂ Architecture

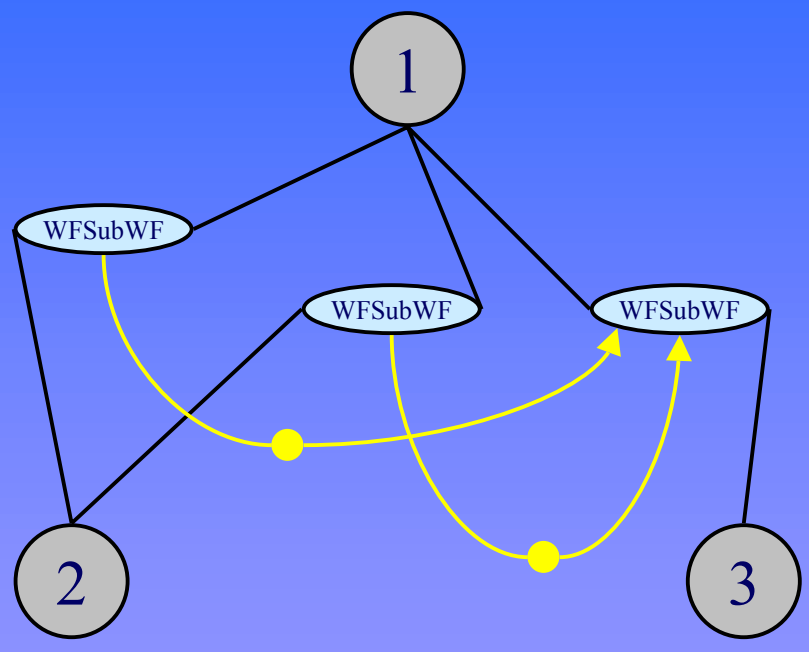
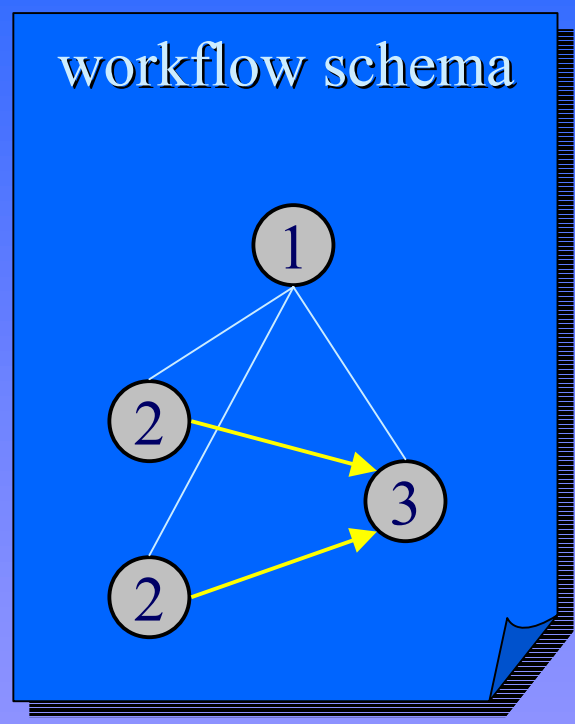


Sample Workflow

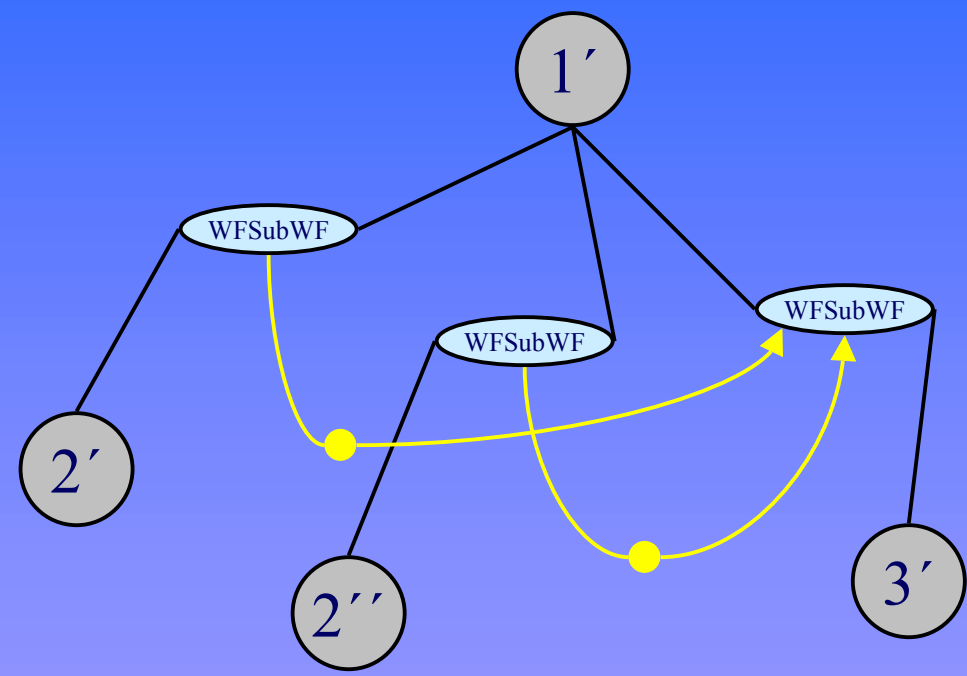
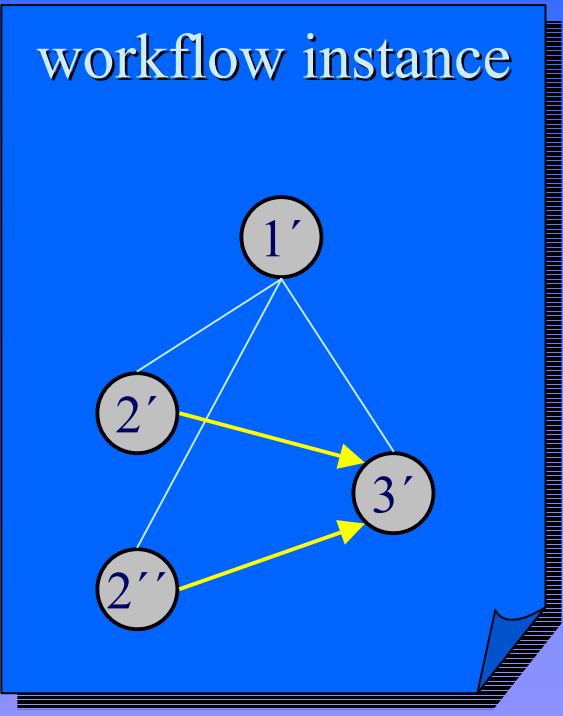
→ control flow
→ data flow



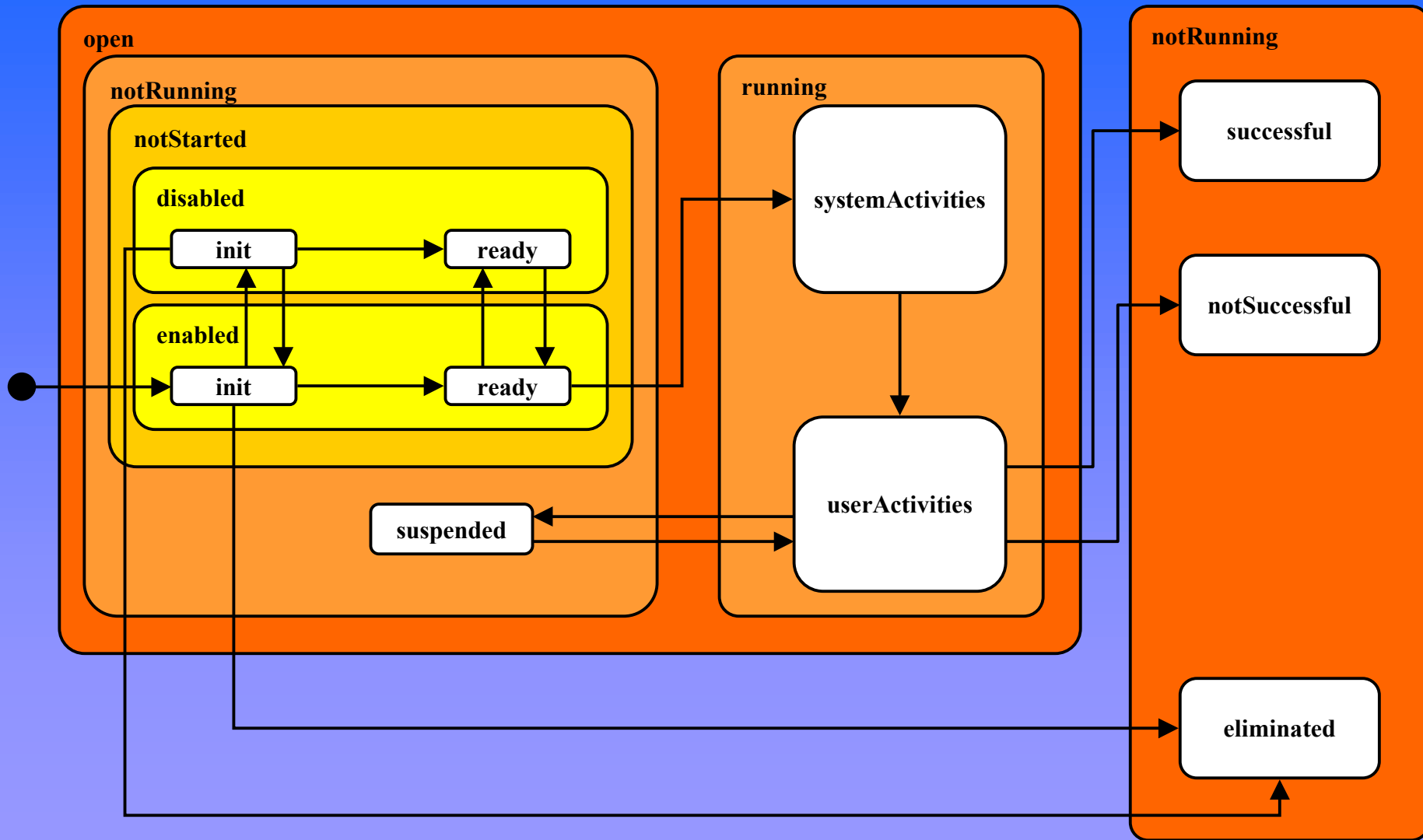
Representation of a Workflow Schema



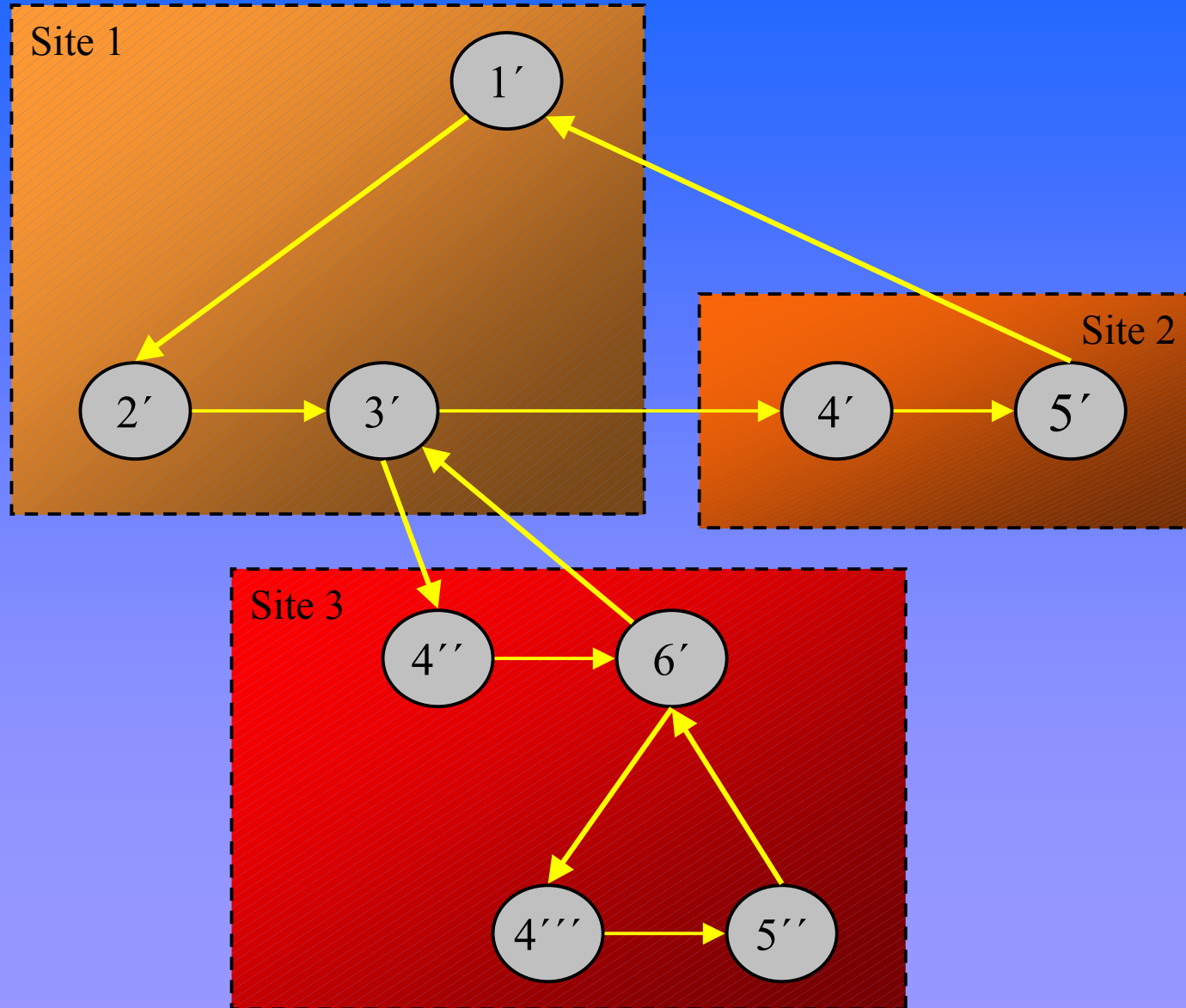
Representation of a Workflow Instance



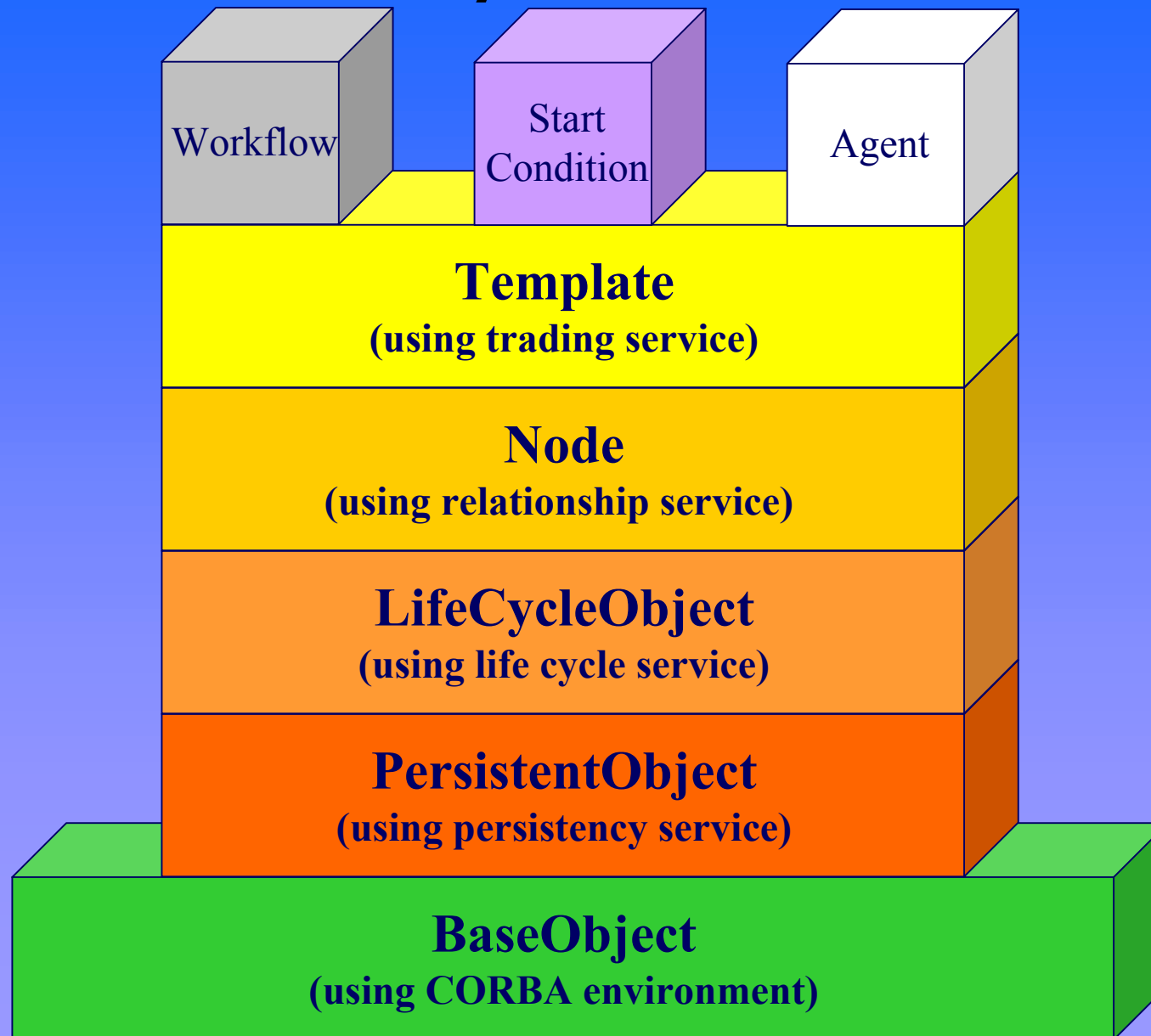
Behavior of Workflow Instance



Distribution Example



Class Hierarchy



IDL for Class Workflow

```
interface Workflow : Template {
    exception InvalidStatetransition { string reason; };

    string    getName();
    void      setName(in string name) raises(CosPersistence::Failed);
    [...]

    Workflow createInstance(in string server,
                           in string machine)
        raises(CosPersistence::Failed, InvalidStatetransition);

    string    getState();

    void      getReady()
        raises(CosPersistence::Failed, InvalidStatetransition);

    void      disable()
        raises(CosPersistence::Failed, InvalidStatetransition);
    void      enable()
        raises(CosPersistence::Failed, InvalidStatetransition);
    [...]
}
```

IDL for Start Conditions

```
interface StartCondition : Template {
    exception UnableToEvaluate {};

    string  getName();

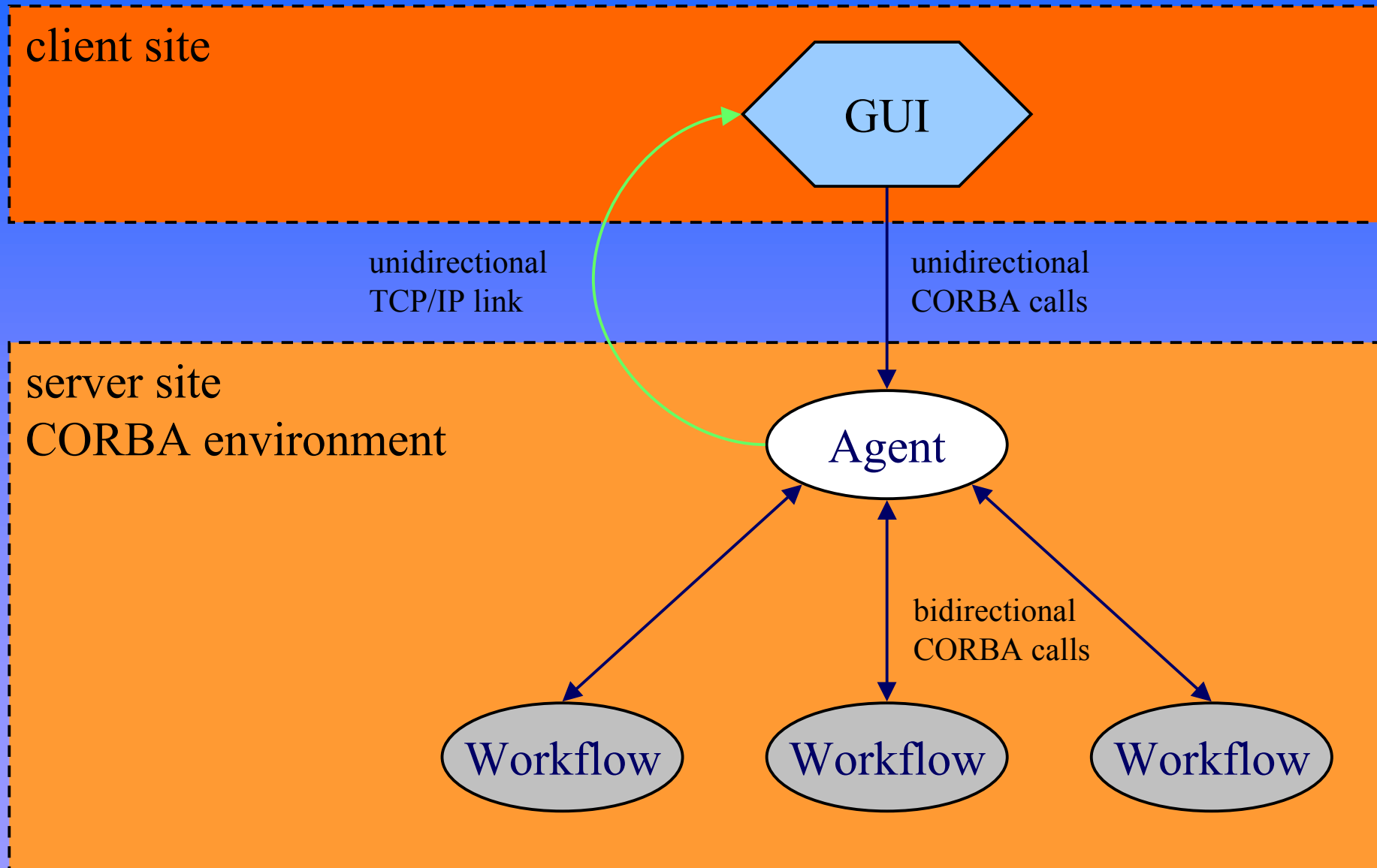
    boolean evaluate(in boolean trigger)
        raises(UnableToEvaluate, CosPersistency::Failed);
    // If trigger = true then the workflow which belongs to this
    // condition will be automatically started or eliminated if
    // possible.
};

interface GenericStartCondition : StartCondition {
    exception SyntaxError {};

    void    setCondition(in string condition)
        raises(CosPersistency::Failed, SyntaxError);

    string  getCondition();
};
```

Communication Model



IDL of Class Agent

```
interface Agent : Template {
    Parameters run_wasa_application(in string classname,
                                   in Parameters input)
        raises(CosPersistence::Failed);

    void write_file(in string filename,
                   in CosPersistence::rawdata data)
        raises(CosPersistence::Failed);

    CosPersistence::rawdata
        read_file(in string filename)
            raises(CosPersistence::Failed);
};
```

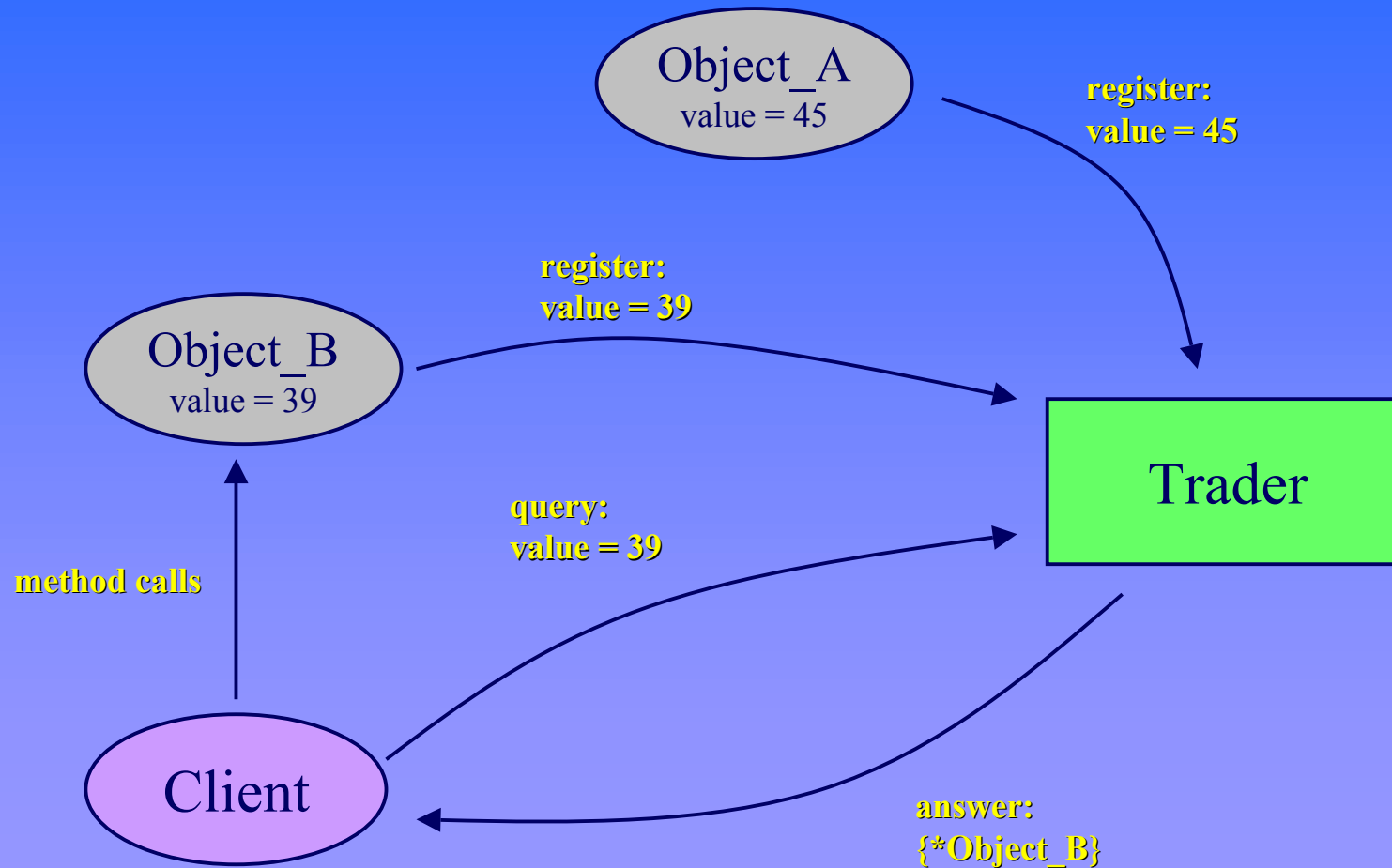
invokes via TCP/IP
communication



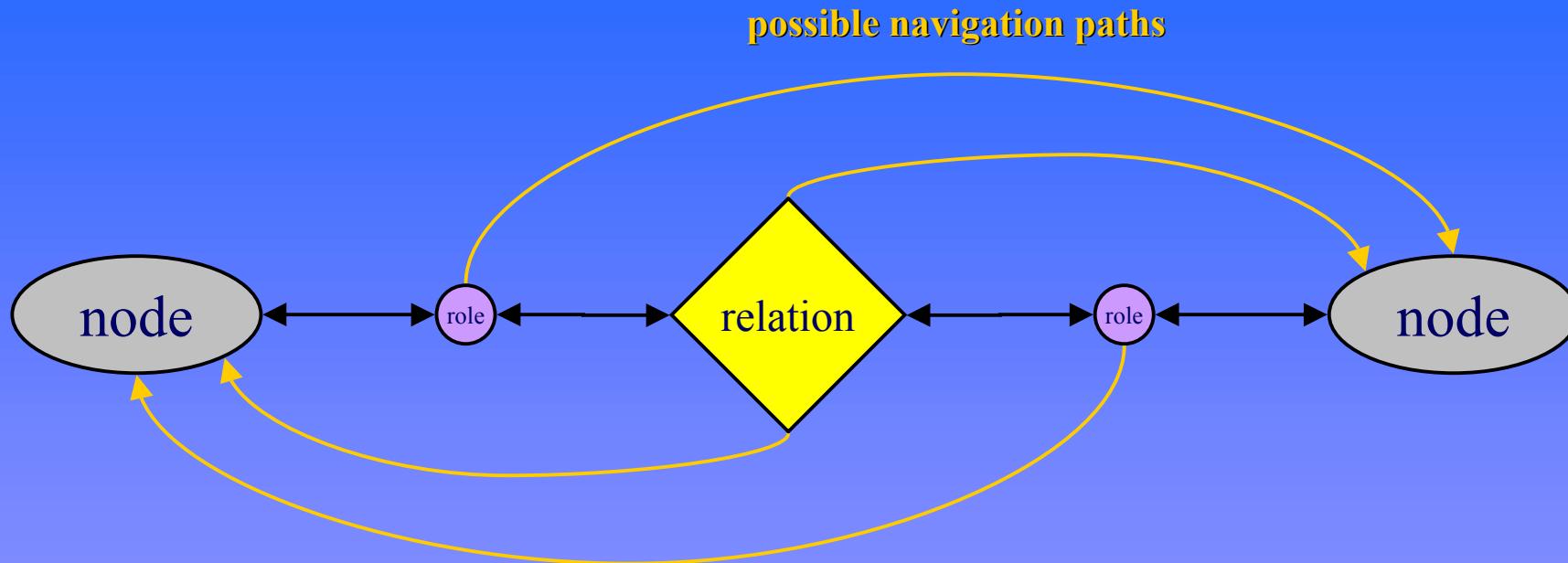
GUI representation of a WASA application (Java):

```
public class WasaApplication implements Runnable {
    public WASA.Parameters execute(WASA.Parameters input);
};
```

Trading Object Service



Relationship Service



all items represented by CORBA objects

associations represented by CORBA references

Life Cycle Service (IDL)

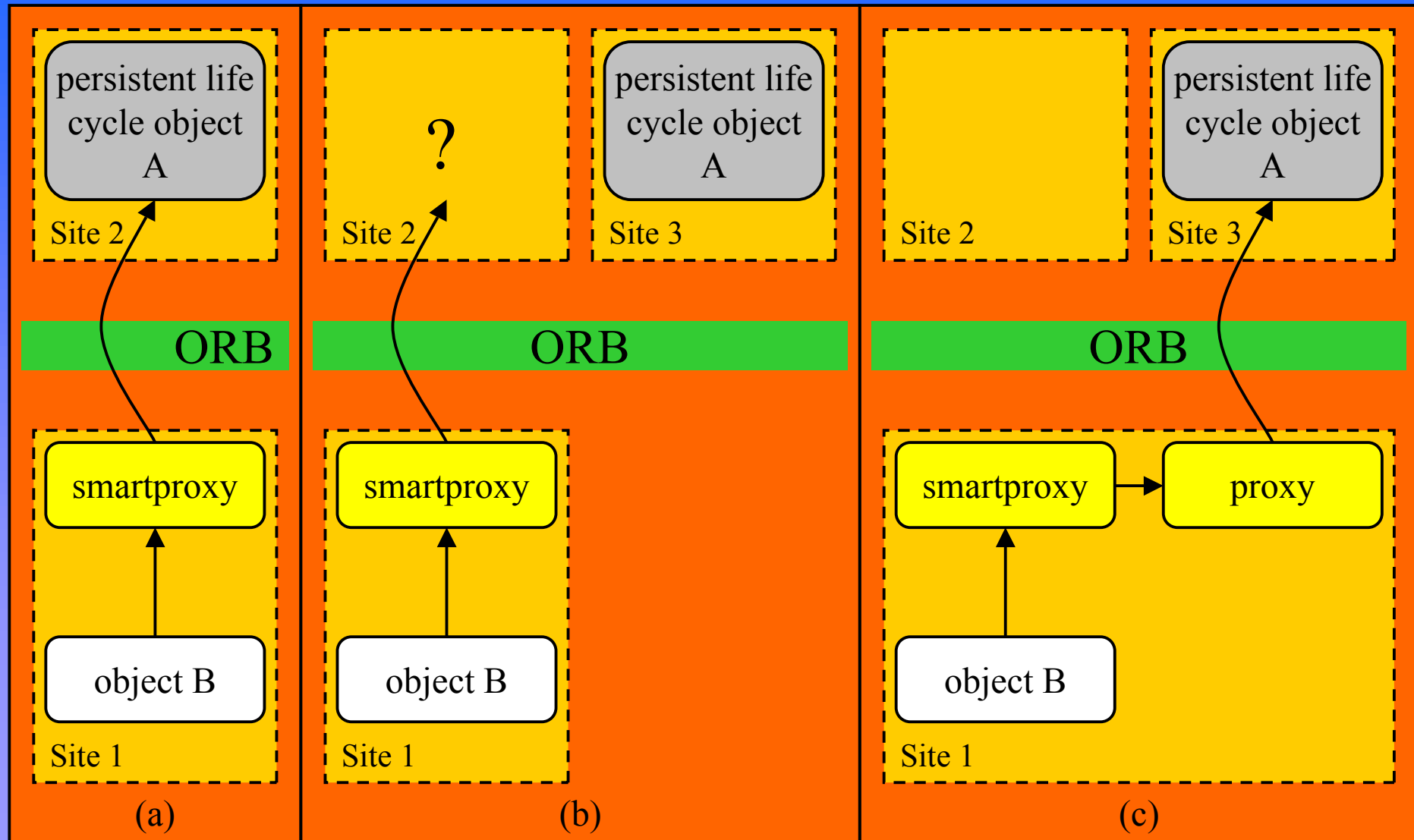
```
interface LifecycleObject {
    LifecycleObject copy(in FactoryFinder there,
                        in Criteria the_criteria)
        raises(NoFactory, NotCopyable,
              InvalidCriteria, CannotMeetCriteria);

    void move(in FactoryFinder there,
              in Criteria the_criteria)
        raises(NoFactory, NotMovable, InvalidCriteria,
              CannotMeetCriteria);

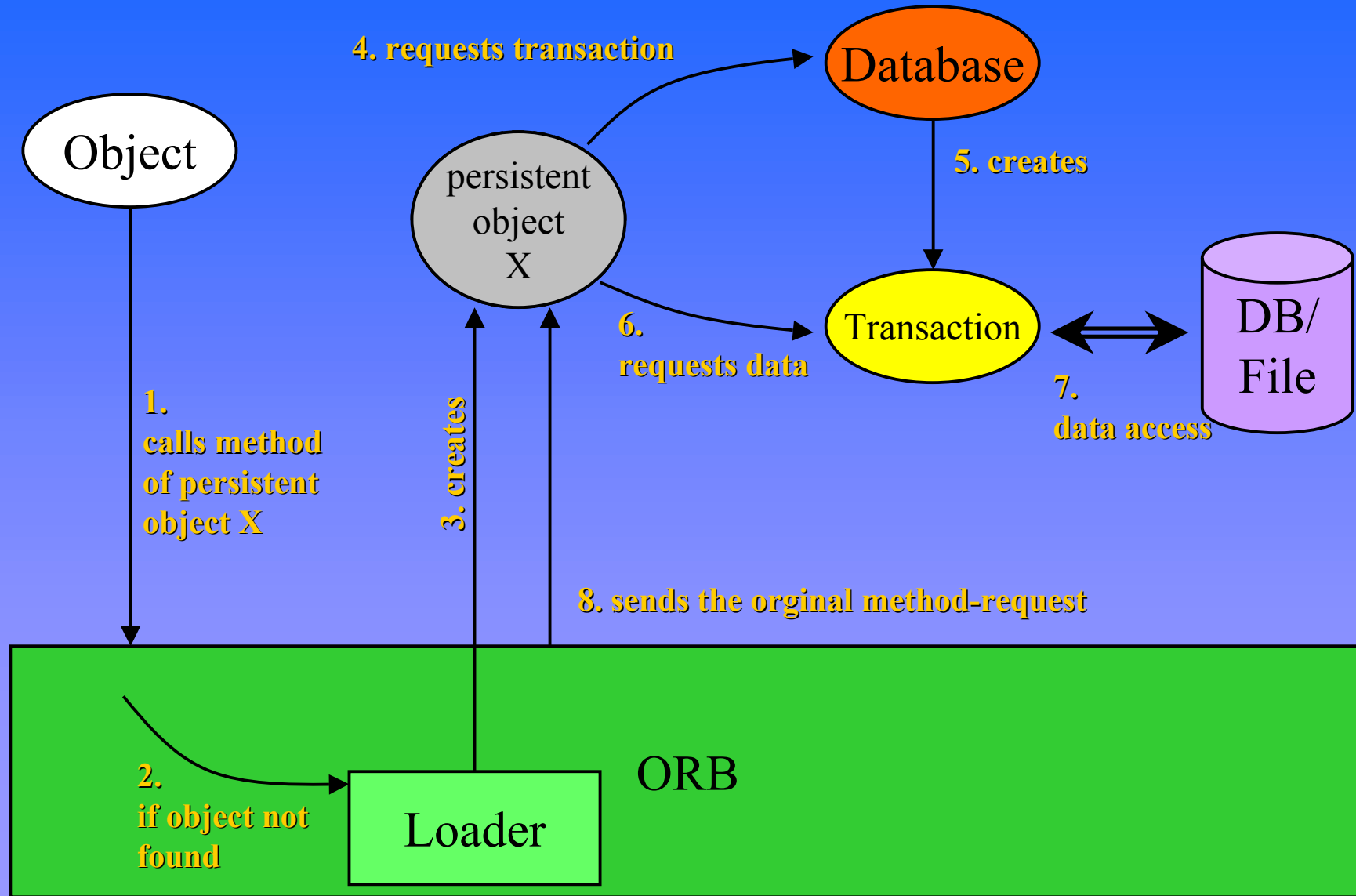
    void remove()
        raises(NotRemoveable);
};
```

```
interface GenericFactory {
    boolean supports(in Key k);
    Object create_object(in Key k,
                        in Criteria the_criteria)
        raises (NoFactory, InvalidCriteria,
              CannotMeetCriteria);
};
```

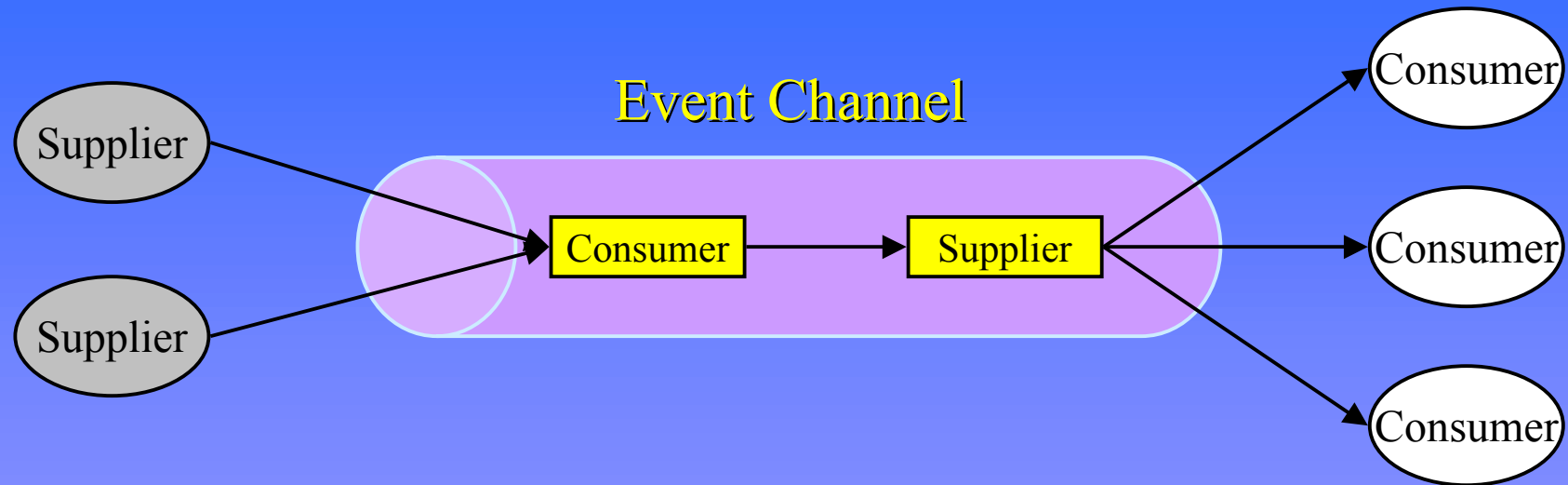
Life Cycle Service



Persistency Service



Event Service



- used for communication with the monitoring tool
- used for role resolution and worklist management

WASA Team

- Coordination
 - Gottfried Vossen
 - Mathias Weske
- Team Members
 - Thomas Serries
 - Dominik Kuropka
 - Jens Hündling
 - Hilmar Schuschel