

```

aspect Observer {

    structure {
        class |Subject {
            + * modify( .. )
            ~ add(|Observer a)
            ~ remove(|Observer a)
            ~ Set<|Observer> getObservers()
        }

        class Set {
            - int size
            ~ Set <|Observer> create()
            ~ add(|Observer)
            ~ remove(|Observer)
            ~ destroy()
        }

        class |Observer {
            + startObserving(|Subject)
            + stopObserving()
            ~ |update(|Subject)
        }

        associations {
            |Observer -> 0..1 |Subject { mySubject }
        }

        instantiation ZeroToMany{
            |Data          -> Subject
            |Associated     -> |Observer
            getAssociated -> getObservers
        }
    }

    def messageView {
        caller = caller : Caller
        target = target: |Observer
        subject = |Subject
    }

    message startObserving {
        caller => target { startObserving( subject ) }
        target => subject { add( target ) }
    }

    message stopObserving {
        caller => target { stopObserving( s ) }
        target => mySubject { remove( target ) }
    }

    message |modify affectedBy notification

    message notification {

```

```

pointcut modifyOperations() {
    caller => target { |modify( .. ) } *
}

// before, after, around?
around() : modifyOperations() {
    target => target { observers := getObservers() }

    loop [ o within observers ] {
        target => o { |update( target ) }
    }
}
}
}

```