MUSE: More USE

Skizze für
ein geplantes, studentisches Projekt

Martin Gogolla
Uni Bremen
AG Datenbanksysteme

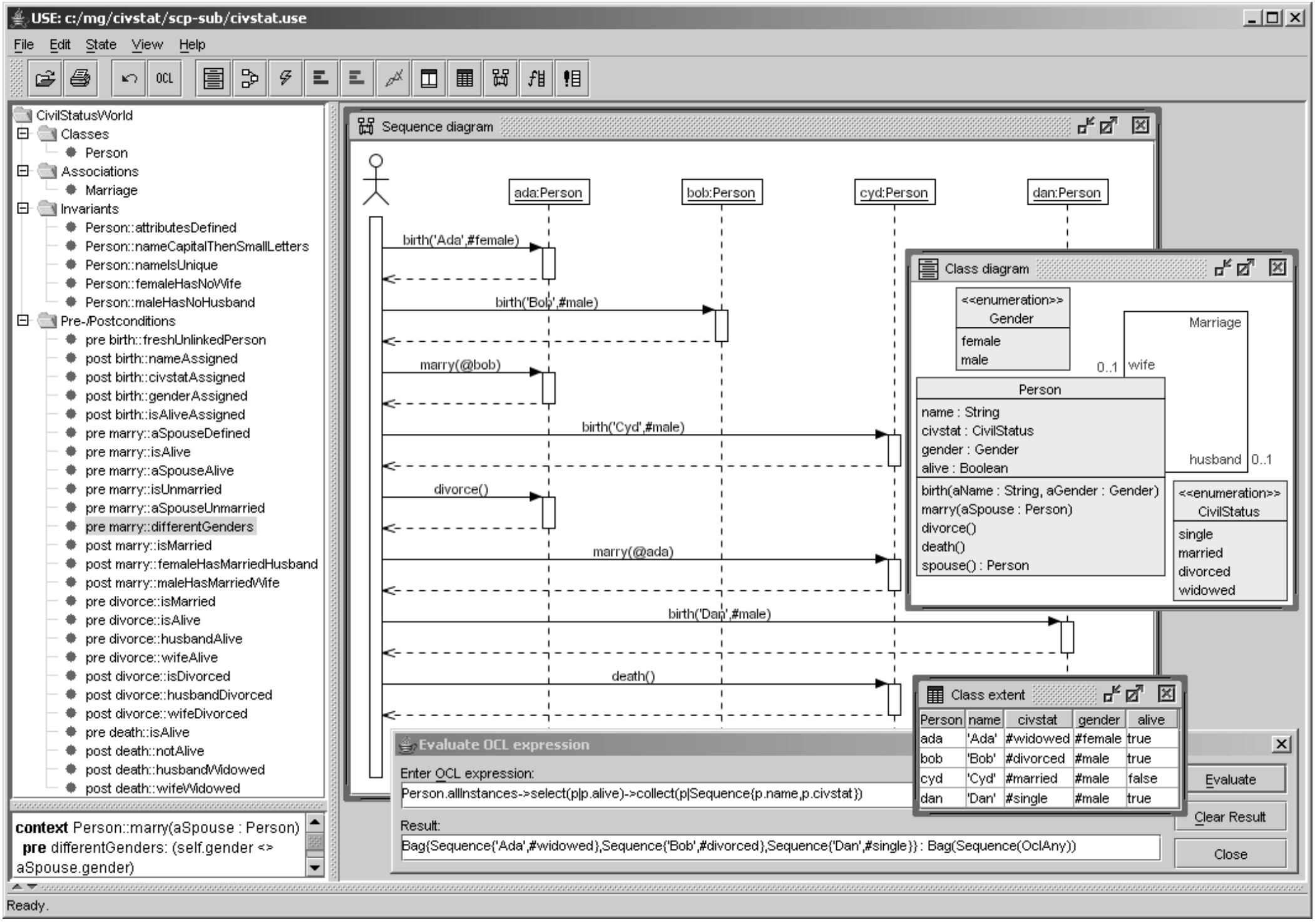
- **Ziel von MUSE (More USE)**
  Einsatz und Weiterentwicklung des Systems USE

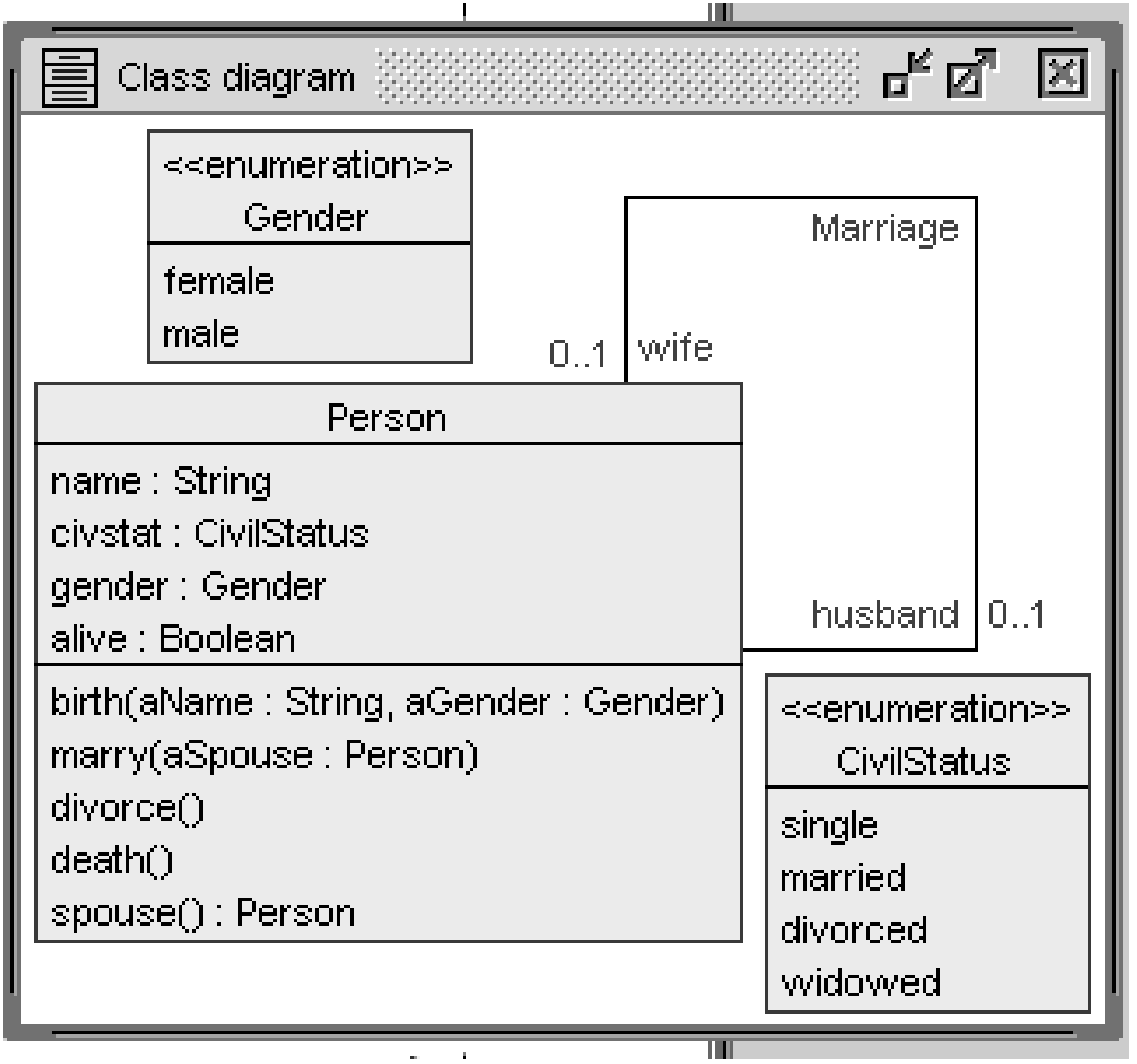
- **USE: UML Specification Environment**
  - **an der Universität Bremen entwickelt in Dissertationen, Diplomarbeiten, Projekten**
  - **dient zur Validation von UML-Beschreibungen**

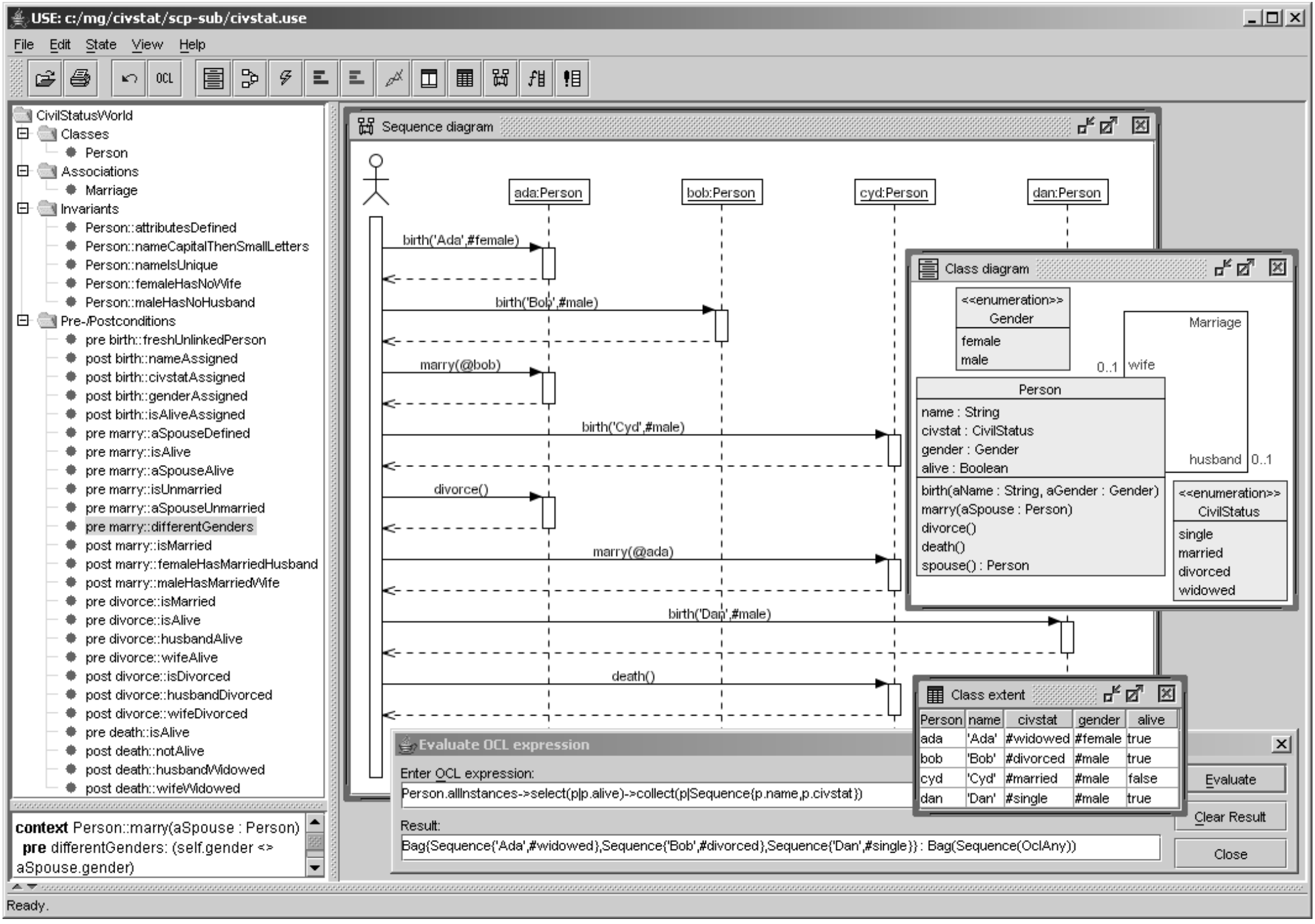
- **in MUSE unterschiedliche Teilprojekte**
  - **empirische Untersuchungen zur unterschiedlichen menschlichen Wahrnehmung diagrammatischer und textueller Beschreibungen**
  - **Entwicklung von Modelltransformationen**
  - **Realisierung weiterer, in USE bisher nicht unterstützter UML Diagrammarten und drei-dimensionaler, audio-visueller Darstellungen**
  - **komplexe Fallstudien**
  - **... weitere ...**

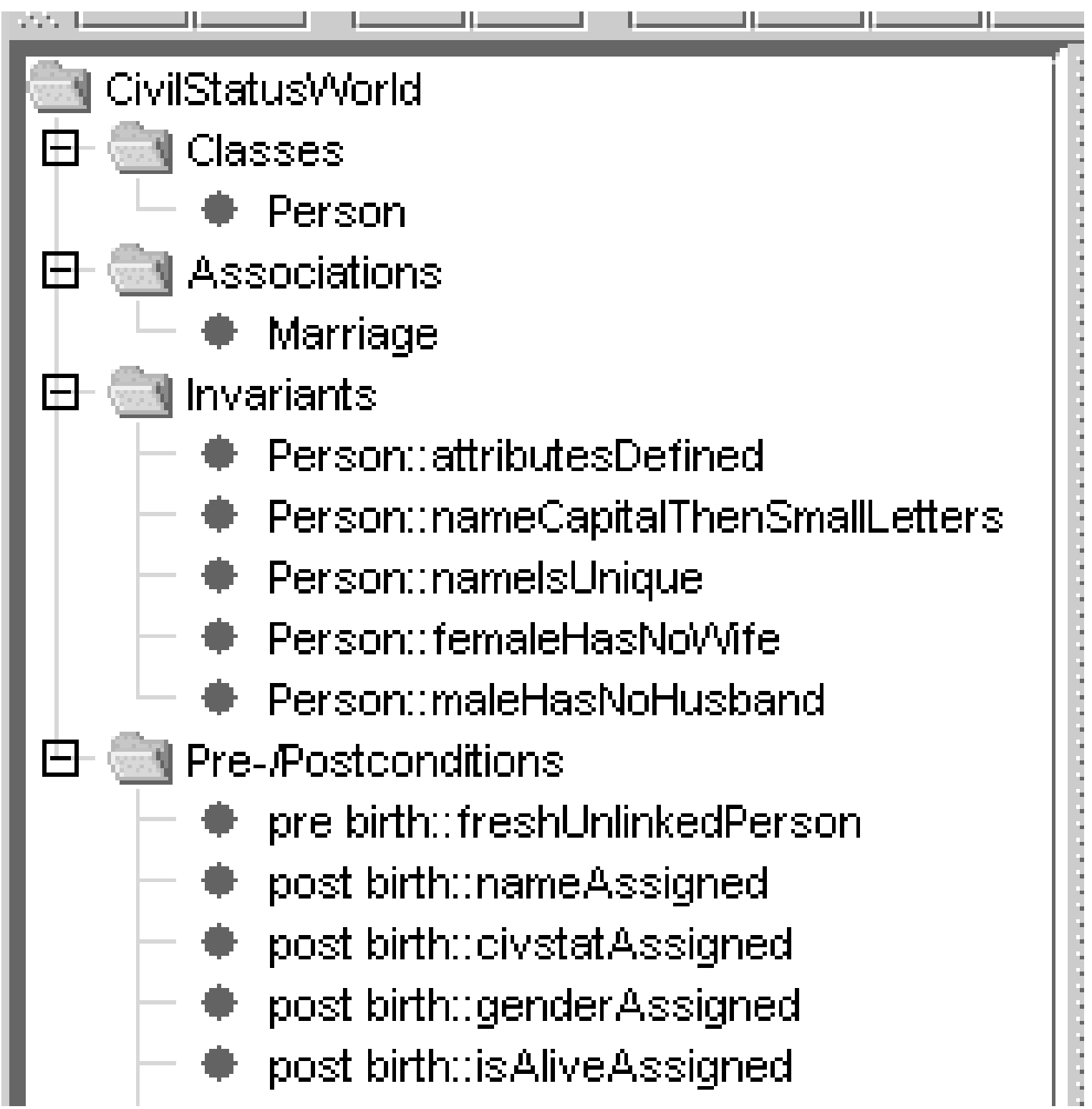
- MUSE offen für Diplom-, Bachelor- und Master-Studierende der Studiengänge Informatik und Digitale Medien

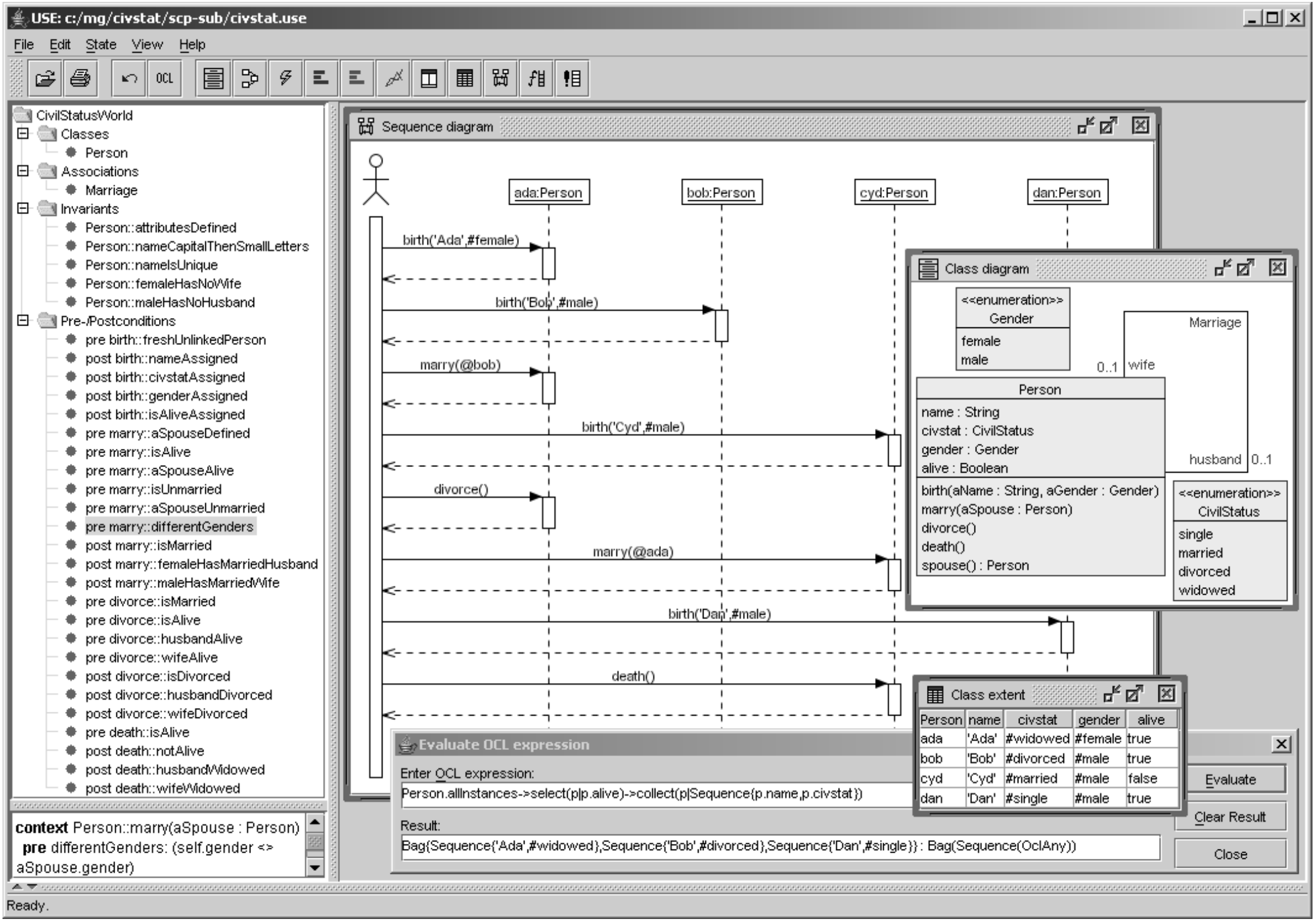
- Anforderungen an Arbeitsstil
  - regelmässige und pünktliche Teilnahme an und Beiträge zu Treffen und -diskussionen
  - wöchentliche Treffen aufgeteilt in Plena und Arbeitsgruppentreffen
  - aktive Beteiligung an Übernahme von Aufgaben
  - Moderation und Dokumentation von Plena und Arbeitsgruppentreffen
  - Beteiligung an (jede/jeder!)
    - Erarbeiten des notwendigen Stoffs (Vortrag, Folien, Ausarbeitung)
    - Problemstellungen und Konzepten für Problemlösungen
    - Lösungen: SW-Entwicklung und Dokumentation
  - pro Semester Aktivitätenbericht (anschliessend Feedback durch Betreuer)

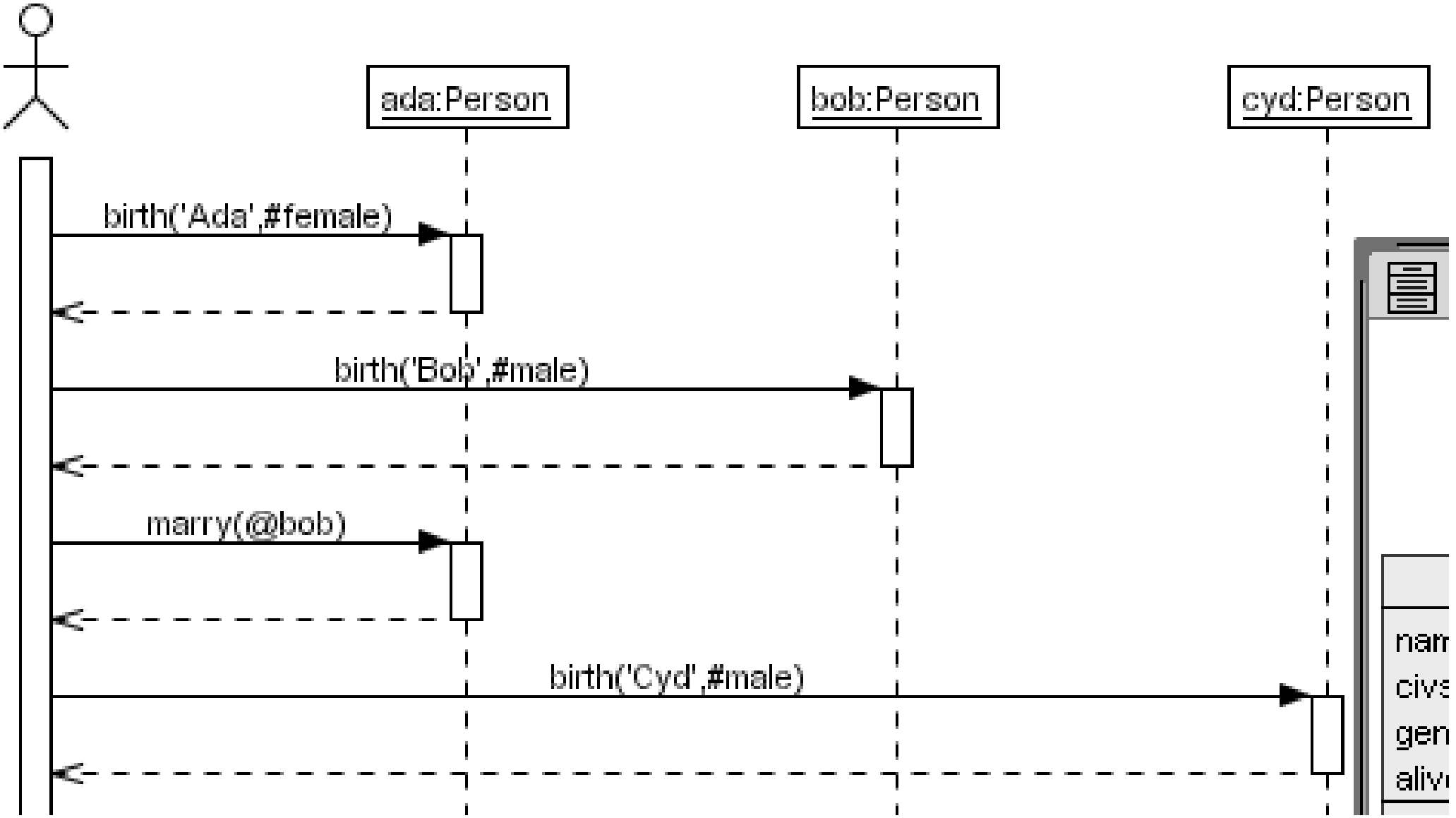

USE: c:/mg/civstat/scp-sub/civstat.use
File Edit State View Help
CivilStatusWorld
Classes
Person
Associations
Marriage
Invariants
Person::attributesDefined
Person::nameCapitalThenSmallLetters
Person::nameIsUnique
Person::femaleHasNoWife
Person::maleHasNoHusband
Pre-/Postconditions
pre birth::freshUnlinkedPerson
post birth::nameAssigned
post birth::civstatAssigned
post birth::genderAssigned
post birth::isAliveAssigned
pre marry::aSpouseDefined
pre marry::isAlive
pre marry::aSpouseAlive
pre marry::isUnmarried
pre marry::aSpouseUnmarried
pre marry::differentGenders
post marry::isMarried
post marry::femaleHasMarriedHusband
post marry::maleHasMarriedWife
pre divorce::isMarried
pre divorce::isAlive
pre divorce::husbandAlive
pre divorce::wifeAlive
post divorce::isDivorced
post divorce::husbandDivorced
post divorce::wifeDivorced
pre death::isAlive
post death::notAlive
post death::husbandWidowed
post death::wifeWidowed
context Person::marry(aSpouse : Person)
pre differentGenders: (self.gender <> aSpouse.gender)
Sequence diagram
ada:Person
bob:Person
cyd:Person
dan:Person
birth('Ada',#female)
birth('Bob',#male)
marry(@bob)
birth('Cyd',#male)
divorce()
marry(@ada)
birth('Dan',#male)
death()
Class diagram
<<enumeration>>
Gender
female
male
Marriage
0..1
wife
husband
0..1
Person
name : String
civstat : CivilStatus
gender : Gender
alive : Boolean
birth(aName : String, aGender : Gender)
marry(aSpouse : Person)
divorce()
death()
spouse() : Person
<<enumeration>>
CivilStatus
single
married
divorced
widowed
Class extent
Person name civstat gender alive
ada 'Ada' #widowed #female true
bob 'Bob' #divorced #male true
cyd 'Cyd' #married #male false
dan 'Dan' #single #male true
Evaluate OCL expression
Enter OCL expression:
Person.allInstances->select(p|p.alive)->collect(p|Sequence{p.name,p.civstat})
Result:
Bag{Sequence{'Ada',#widowed},Sequence{'Bob',#divorced},Sequence{'Dan',#single}} : Bag(Sequence(OclAny))
Evaluate
Clear Result
Close
Ready.

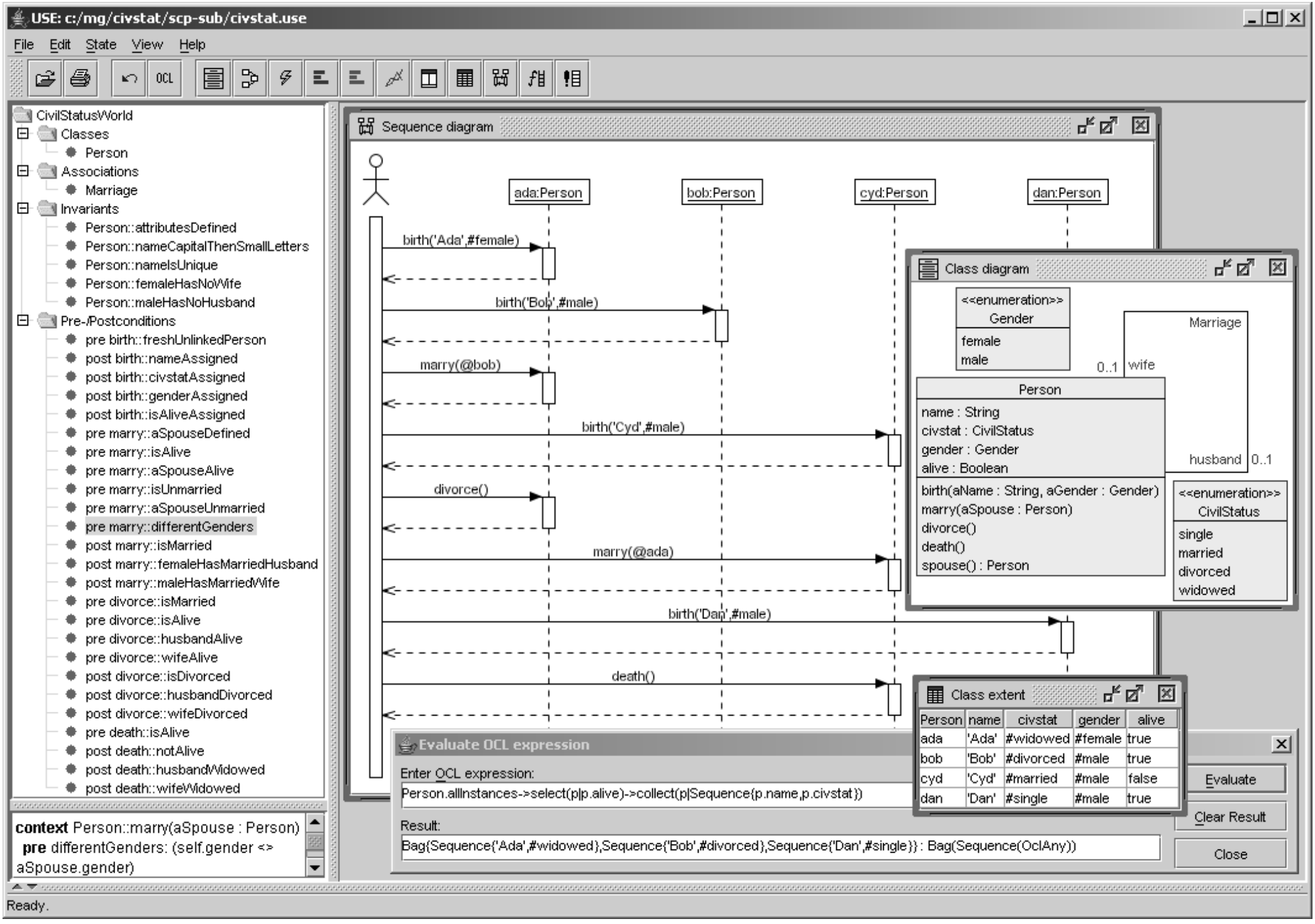

Class diagram
<<enumeration>>
Gender
female
male
Marriage
0..1
wife
husband
0..1
Person
name : String
civstat : CivilStatus
gender : Gender
alive : Boolean
birth(aName : String, aGender : Gender)
marry(aSpouse : Person)
divorce()
death()
spouse() : Person
<<enumeration>>
CivilStatus
single
married
divorced
widowed

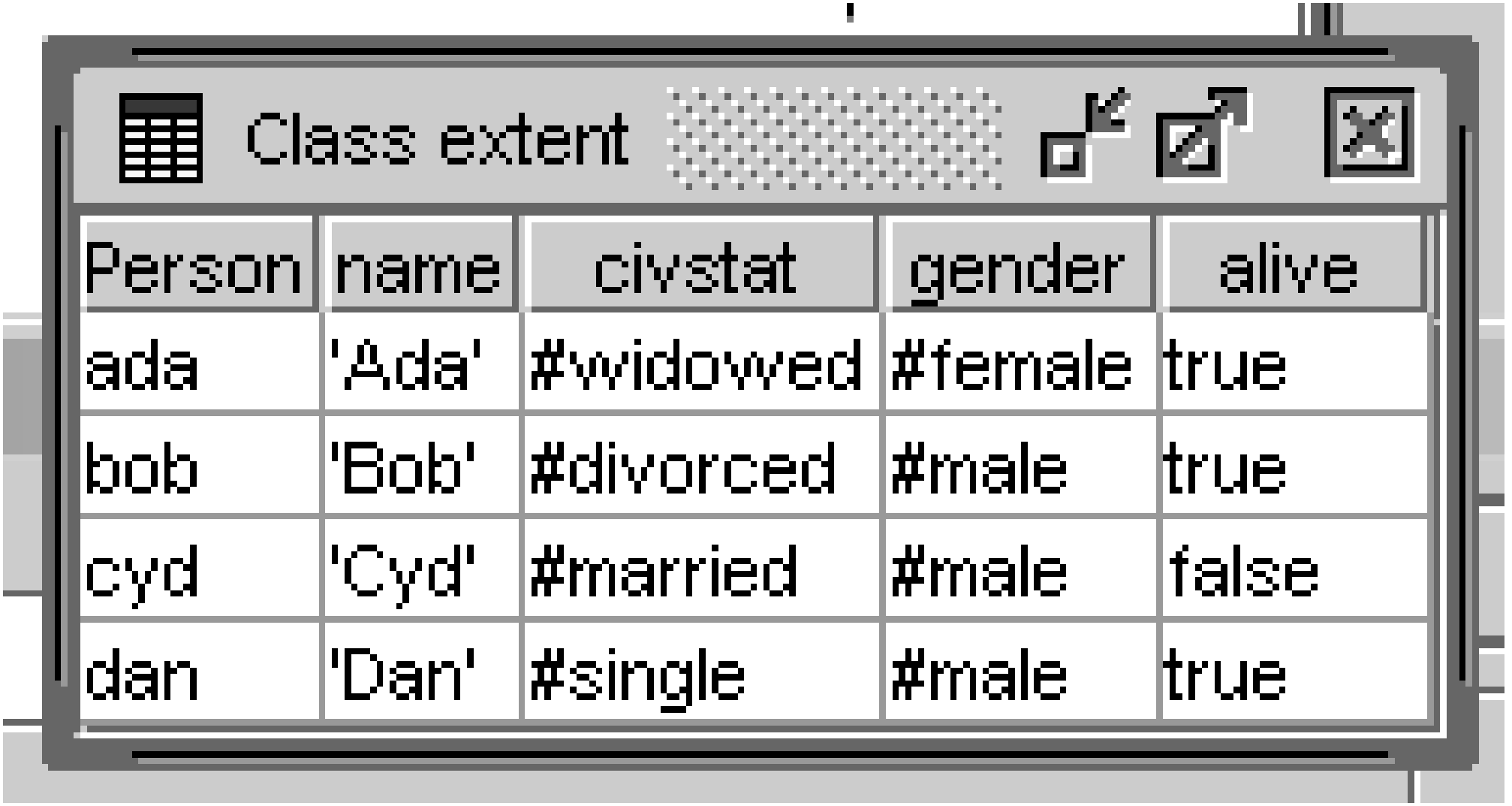

USE: c:/mg/civstat/scp-sub/civstat.use
File Edit State View Help
CivilStatusWorld
Classes
Person
Associations
Marriage
Invariants
Person::attributesDefined
Person::nameCapitalThenSmallLetters
Person::nameIsUnique
Person::femaleHasNoWife
Person::maleHasNoHusband
Pre-/Postconditions
pre birth::freshUnlinkedPerson
post birth::nameAssigned
post birth::civstatAssigned
post birth::genderAssigned
post birth::isAliveAssigned
pre marry::aSpouseDefined
pre marry::isAlive
pre marry::aSpouseAlive
pre marry::isUnmarried
pre marry::aSpouseUnmarried
pre marry::differentGenders
post marry::isMarried
post marry::femaleHasMarriedHusband
post marry::maleHasMarriedWife
pre divorce::isMarried
pre divorce::isAlive
pre divorce::husbandAlive
pre divorce::wifeAlive
post divorce::isDivorced
post divorce::husbandDivorced
post divorce::wifeDivorced
pre death::isAlive
post death::notAlive
post death::husbandWidowed
post death::wifeWidowed
context Person::marry(aSpouse : Person)
pre differentGenders: (self.gender <> aSpouse.gender)
Sequence diagram
ada:Person
bob:Person
cyd:Person
dan:Person
birth('Ada',#female)
birth('Bob',#male)
marry(@bob)
birth('Cyd',#male)
divorce()
marry(@ada)
birth('Dan',#male)
death()
Class diagram
<<enumeration>>
Gender
female
male
Marriage
0..1
wife
husband
0..1
Person
name : String
civstat : CivilStatus
gender : Gender
alive : Boolean
birth(aName : String, aGender : Gender)
marry(aSpouse : Person)
divorce()
death()
spouse() : Person
<<enumeration>>
CivilStatus
single
married
divorced
widowed
Class extent
Person name civstat gender alive
ada 'Ada' #widowed #female true
bob 'Bob' #divorced #male true
cyd 'Cyd' #married #male false
dan 'Dan' #single #male true
Evaluate OCL expression
Enter OCL expression:
Person.allInstances->select(p|p.alive)->collect(p|Sequence{p.name,p.civstat})
Result:
Bag{Sequence{'Ada',#widowed},Sequence{'Bob',#divorced},Sequence{'Dan',#single}} : Bag(Sequence(OclAny))
Evaluate
Clear Result
Close
Ready.


CivilStatusWorld
Classes
Person
Associations
Marriage
Invariants
Person::attributesDefined
Person::nameCapitalThenSmallLetters
Person::nameIsUnique
Person::femaleHasNoWife
Person::maleHasNoHusband
Pre-/Postconditions
pre birth::freshUnlinkedPerson
post birth::nameAssigned
post birth::civstatAssigned
post birth::genderAssigned
post birth::isAliveAssigned

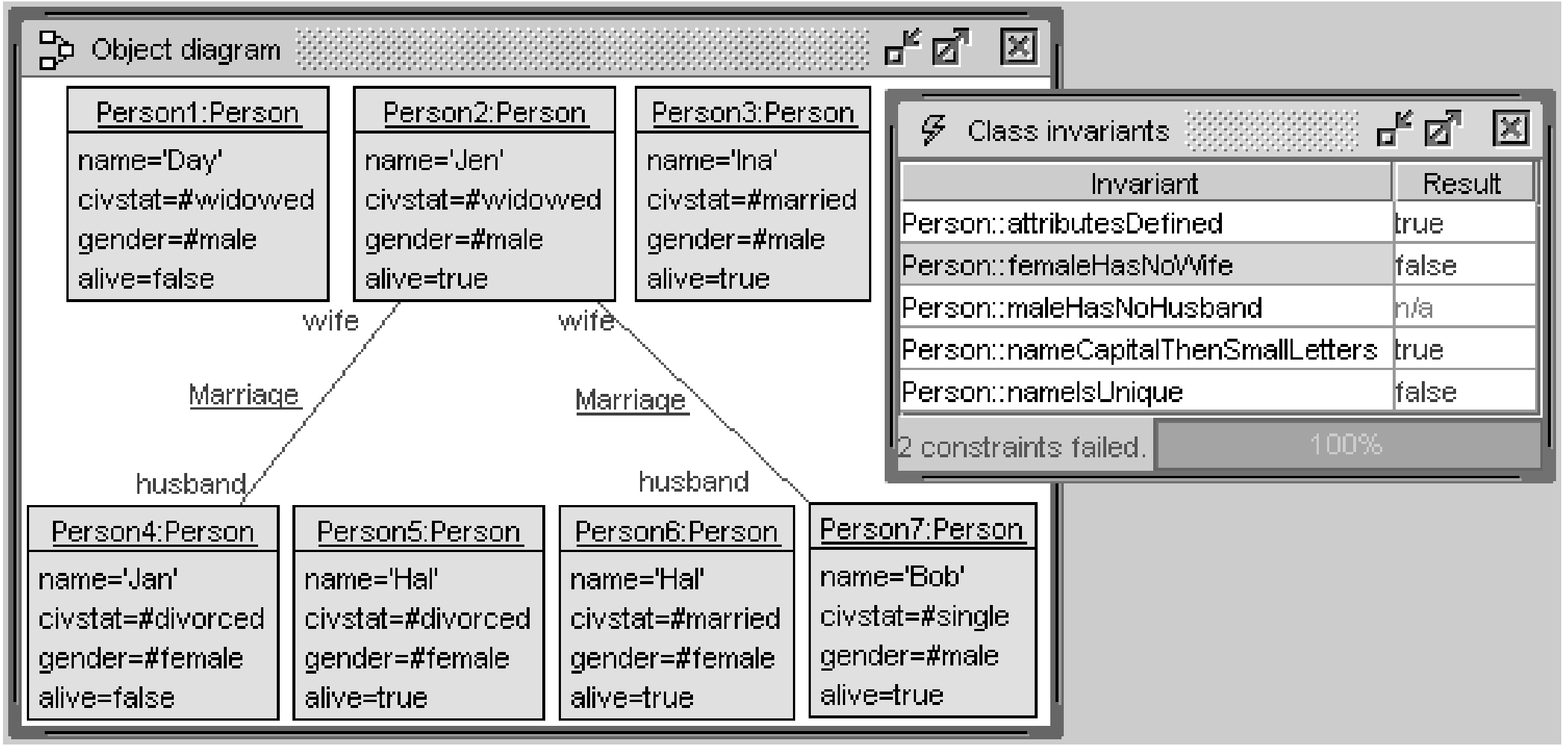

USE: c:/mg/civstat/scp-sub/civstat.use
File Edit State View Help
CivilStatusWorld
Classes
Person
Associations
Marriage
Invariants
Person::attributesDefined
Person::nameCapitalThenSmallLetters
Person::nameIsUnique
Person::femaleHasNoWife
Person::maleHasNoHusband
Pre-/Postconditions
pre birth::freshUnlinkedPerson
post birth::nameAssigned
post birth::civstatAssigned
post birth::genderAssigned
post birth::isAliveAssigned
pre marry::aSpouseDefined
pre marry::isAlive
pre marry::aSpouseAlive
pre marry::isUnmarried
pre marry::aSpouseUnmarried
pre marry::differentGenders
post marry::isMarried
post marry::femaleHasMarriedHusband
post marry::maleHasMarriedWife
pre divorce::isMarried
pre divorce::isAlive
pre divorce::husbandAlive
pre divorce::wifeAlive
post divorce::isDivorced
post divorce::husbandDivorced
post divorce::wifeDivorced
pre death::isAlive
post death::notAlive
post death::husbandWidowed
post death::wifeWidowed
context Person::marry(aSpouse : Person)
pre differentGenders: (self.gender <> aSpouse.gender)
Sequence diagram
ada:Person
bob:Person
cyd:Person
dan:Person
birth('Ada',#female)
birth('Bob',#male)
marry(@bob)
birth('Cyd',#male)
divorce()
marry(@ada)
birth('Dan',#male)
death()
Class diagram
<<enumeration>>
Gender
female
male
Marriage
0..1
wife
husband
0..1
Person
name : String
civstat : CivilStatus
gender : Gender
alive : Boolean
birth(aName : String, aGender : Gender)
marry(aSpouse : Person)
divorce()
death()
spouse() : Person
<<enumeration>>
CivilStatus
single
married
divorced
widowed
Class extent
Person name civstat gender alive
ada 'Ada' #widowed #female true
bob 'Bob' #divorced #male true
cyd 'Cyd' #married #male false
dan 'Dan' #single #male true
Evaluate OCL expression
Enter OCL expression:
Person.allInstances->select(p|p.alive)->collect(p|Sequence{p.name,p.civstat})
Result:
Bag{Sequence{'Ada',#widowed},Sequence{'Bob',#divorced},Sequence{'Dan',#single}} : Bag(Sequence(OclAny))
Evaluate
Clear Result
Close
Ready.

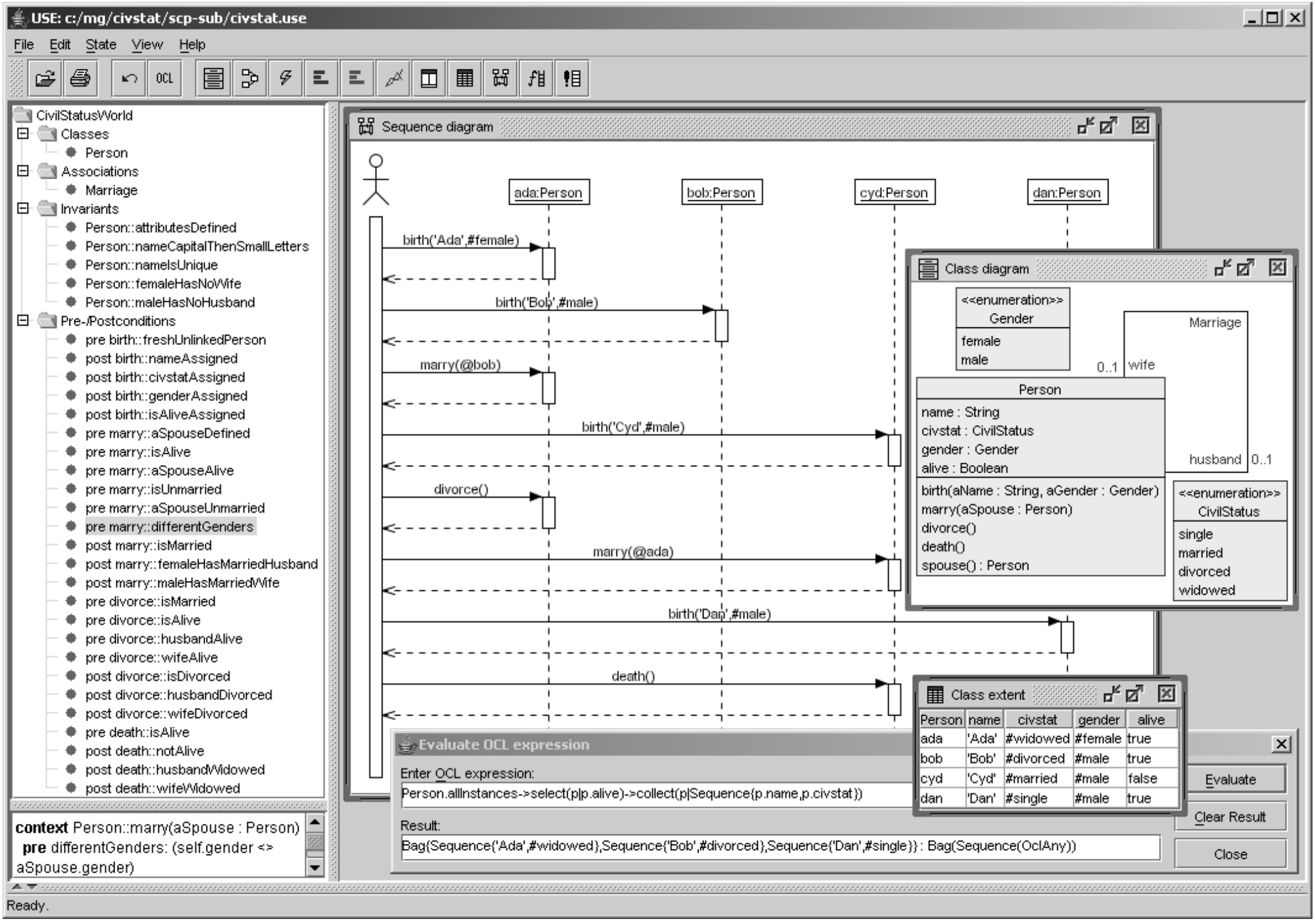

ada:Person
bob:Person
cyd:Person
birth('Ada',#female)
birth('Bob',#male)
marry(@bob)
birth('Cyd',#male)

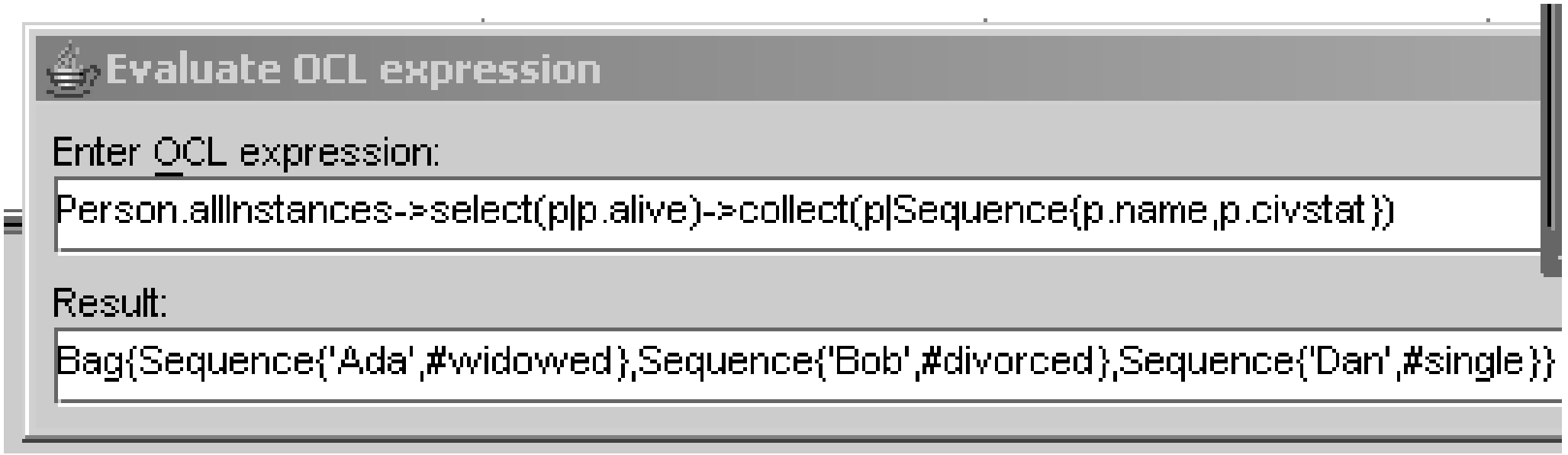

USE: c:/mg/civstat/scp-sub/civstat.use
File Edit State View Help
CivilStatusWorld
Classes
Person
Associations
Marriage
Invariants
Person::attributesDefined
Person::nameCapitalThenSmallLetters
Person::nameIsUnique
Person::femaleHasNoWife
Person::maleHasNoHusband
Pre-/Postconditions
pre birth::freshUnlinkedPerson
post birth::nameAssigned
post birth::civstatAssigned
post birth::genderAssigned
post birth::isAliveAssigned
pre marry::aSpouseDefined
pre marry::isAlive
pre marry::aSpouseAlive
pre marry::isUnmarried
pre marry::aSpouseUnmarried
pre marry::differentGenders
post marry::isMarried
post marry::femaleHasMarriedHusband
post marry::maleHasMarriedWife
pre divorce::isMarried
pre divorce::isAlive
pre divorce::husbandAlive
pre divorce::wifeAlive
post divorce::isDivorced
post divorce::husbandDivorced
post divorce::wifeDivorced
pre death::isAlive
post death::notAlive
post death::husbandWidowed
post death::wifeWidowed
context Person::marry(aSpouse : Person)
pre differentGenders: (self.gender <> aSpouse.gender)
Sequence diagram
ada:Person
bob:Person
cyd:Person
dan:Person
birth('Ada',#female)
birth('Bob',#male)
marry(@bob)
birth('Cyd',#male)
divorce()
marry(@ada)
birth('Dan',#male)
death()
Class diagram
<<enumeration>>
Gender
female
male
Marriage
0..1
wife
husband
0..1
Person
name : String
civstat : CivilStatus
gender : Gender
alive : Boolean
birth(aName : String, aGender : Gender)
marry(aSpouse : Person)
divorce()
death()
spouse() : Person
<<enumeration>>
CivilStatus
single
married
divorced
widowed
Class extent
Person name civstat gender alive
ada 'Ada' #widowed #female true
bob 'Bob' #divorced #male true
cyd 'Cyd' #married #male false
dan 'Dan' #single #male true
Evaluate OCL expression
Enter OCL expression:
Person.allInstances->select(p|p.alive)->collect(p|Sequence{p.name,p.civstat})
Result:
Bag{Sequence{'Ada',#widowed},Sequence{'Bob',#divorced},Sequence{'Dan',#single}} : Bag(Sequence(OclAny))
Evaluate
Clear Result
Close
Ready.

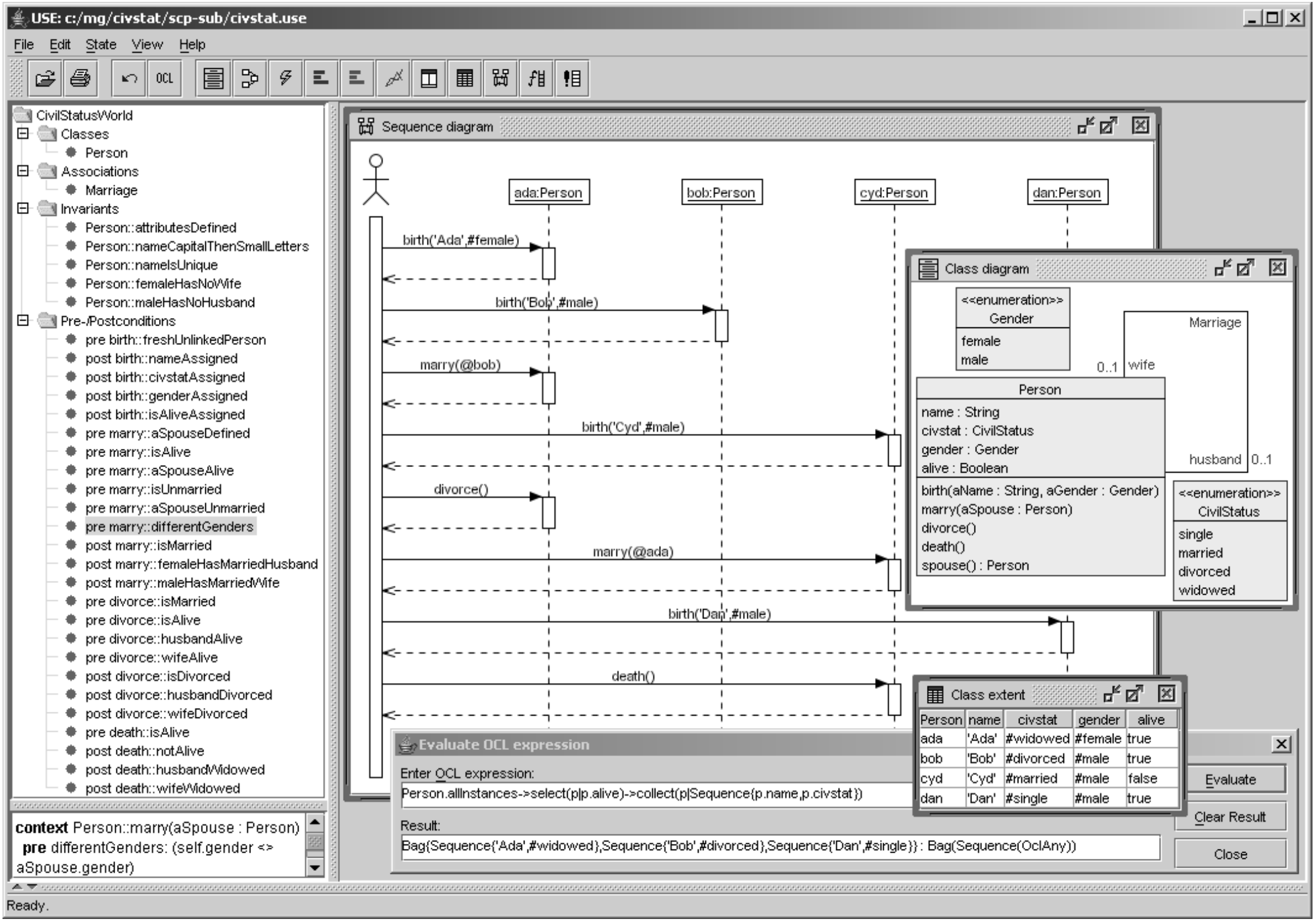


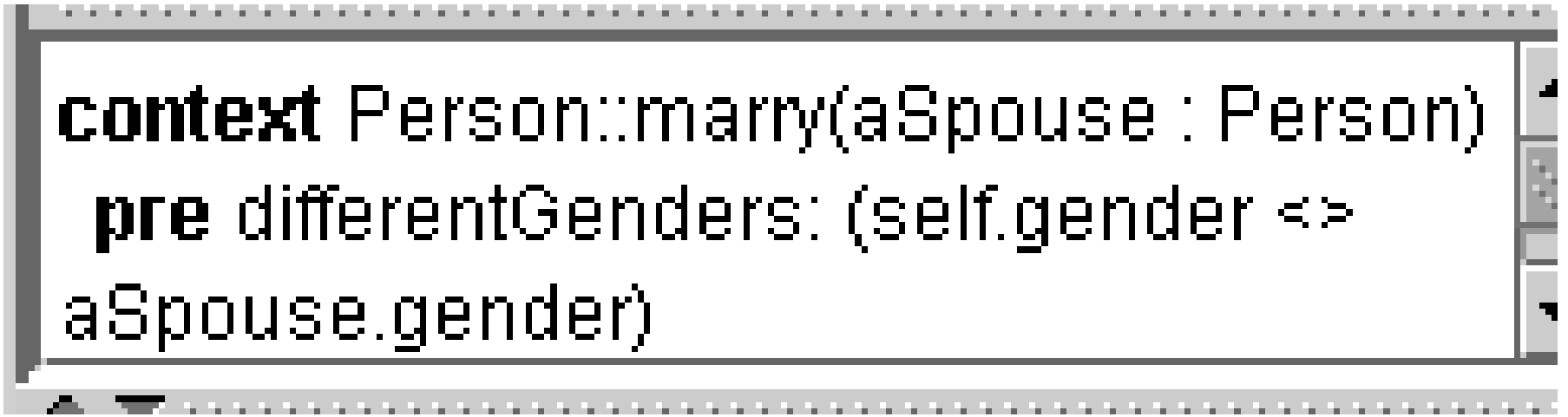
Person	name	civstat	gender	alive
ada	'Ada'	#widowed	#female	true
bob	'Bob'	#divorced	#male	true
cyd	'Cyd'	#married	#male	false
dan	'Dan'	#single	#male	true

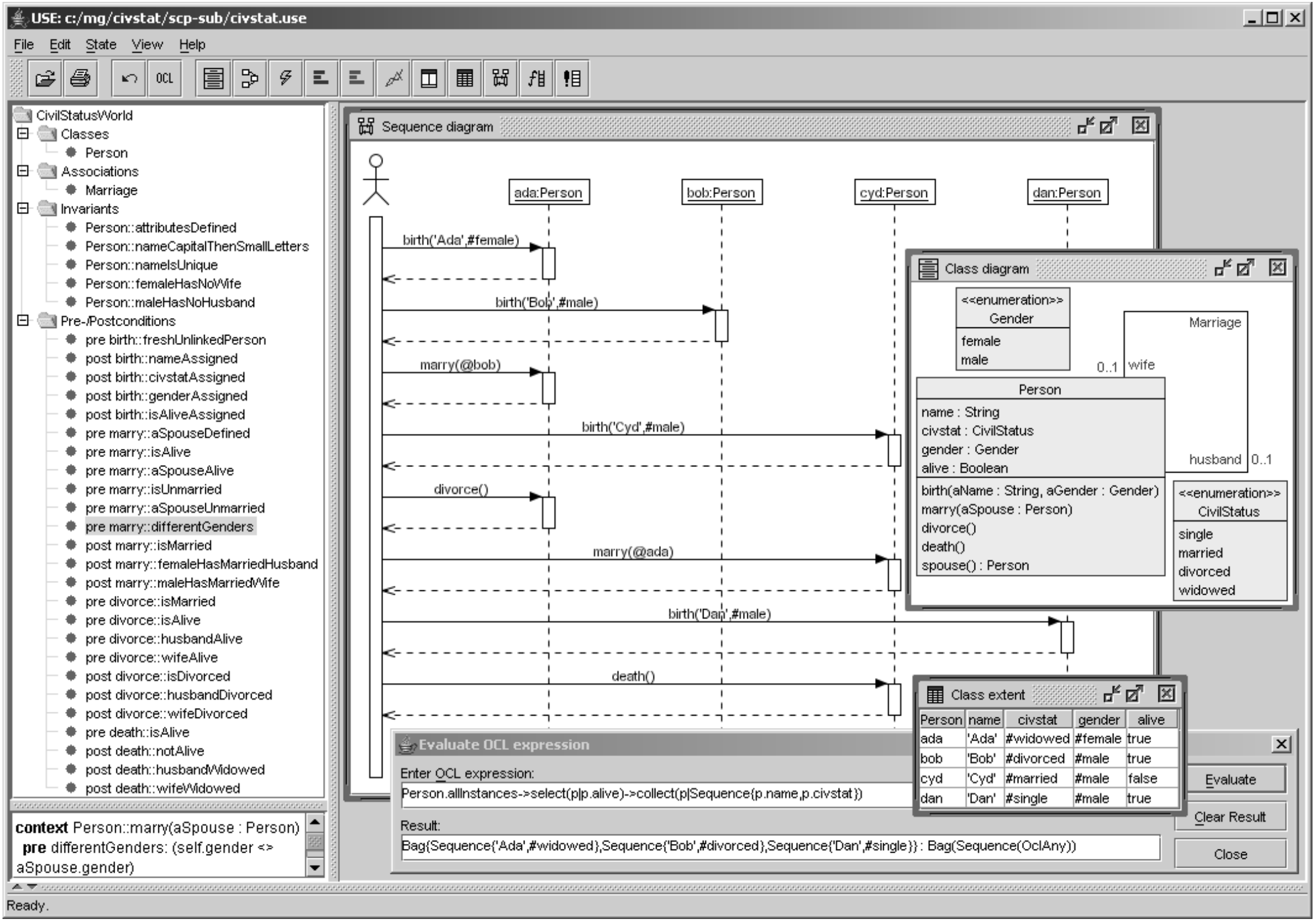

Object diagram
Person1:Person
name='Day'
civstat=#widowed
gender=#male
alive=false
Person2:Person
name='Jen'
civstat=#widowed
gender=#male
alive=true
Person3:Person
name='Ina'
civstat=#married
gender=#male
alive=true
wife
wife
Marriage
Marriage
husband
husband
Person4:Person
name='Jan'
civstat=#divorced
gender=#female
alive=false
Person5:Person
name='Hal'
civstat=#divorced
gender=#female
alive=true
Person6:Person
name='Hal'
civstat=#married
gender=#female
alive=true
Person7:Person
name='Bob'
civstat=#single
gender=#male
alive=true
Class invariants
Invariant
Result
Person::attributesDefined
true
Person::femaleHasNoWife
false
Person::maleHasNoHusband
n/a
Person::nameCapitalThenSmallLetters
true
Person::nameIsUnique
false
2 constraints failed.
100%


USE: c:/mg/civstat/scp-sub/civstat.use
File Edit State View Help
CivilStatusWorld
Classes
Person
Associations
Marriage
Invariants
Person::attributesDefined
Person::nameCapitalThenSmallLetters
Person::nameIsUnique
Person::femaleHasNoWife
Person::maleHasNoHusband
Pre-/Postconditions
pre birth::freshUnlinkedPerson
post birth::nameAssigned
post birth::civstatAssigned
post birth::genderAssigned
post birth::isAliveAssigned
pre marry::aSpouseDefined
pre marry::isAlive
pre marry::aSpouseAlive
pre marry::isUnmarried
pre marry::aSpouseUnmarried
pre marry::differentGenders
post marry::isMarried
post marry::femaleHasMarriedHusband
post marry::maleHasMarriedWife
pre divorce::isMarried
pre divorce::isAlive
pre divorce::husbandAlive
pre divorce::wifeAlive
post divorce::isDivorced
post divorce::husbandDivorced
post divorce::wifeDivorced
pre death::isAlive
post death::notAlive
post death::husbandWidowed
post death::wifeWidowed
context Person::marry(aSpouse : Person)
pre differentGenders: (self.gender <> aSpouse.gender)
Sequence diagram
ada:Person
bob:Person
cyd:Person
dan:Person
birth('Ada',#female)
birth('Bob',#male)
marry(@bob)
birth('Cyd',#male)
divorce()
marry(@ada)
birth('Dan',#male)
death()
Class diagram
<<enumeration>>
Gender
female
male
Marriage
0..1
wife
husband
0..1
Person
name : String
civstat : CivilStatus
gender : Gender
alive : Boolean
birth(aName : String, aGender : Gender)
marry(aSpouse : Person)
divorce()
death()
spouse() : Person
<<enumeration>>
CivilStatus
single
married
divorced
widowed
Class extent
Person name civstat gender alive
ada 'Ada' #widowed #female true
bob 'Bob' #divorced #male true
cyd 'Cyd' #married #male false
dan 'Dan' #single #male true
Evaluate OCL expression
Enter OCL expression:
Person.allInstances->select(p|p.alive)->collect(p|Sequence{p.name,p.civstat})
Result:
Bag{Sequence{'Ada',#widowed},Sequence{'Bob',#divorced},Sequence{'Dan',#single}} : Bag(Sequence(OclAny))
Evaluate
Clear Result
Close
Ready.


Evaluate OCL expression
Enter OCL expression:
Person.allInstances->select(p|p.alive)->collect(p|Sequence{p.name,p.civstat})
Result:
Bag{Sequence{'Ada',#widowed},Sequence{'Bob',#divorced},Sequence{'Dan',#single}}


USE: c:/mg/civstat/scp-sub/civstat.use
File Edit State View Help
CivilStatusWorld
Classes
Person
Associations
Marriage
Invariants
Person::attributesDefined
Person::nameCapitalThenSmallLetters
Person::nameIsUnique
Person::femaleHasNoWife
Person::maleHasNoHusband
Pre-/Postconditions
pre birth::freshUnlinkedPerson
post birth::nameAssigned
post birth::civstatAssigned
post birth::genderAssigned
post birth::isAliveAssigned
pre marry::aSpouseDefined
pre marry::isAlive
pre marry::aSpouseAlive
pre marry::isUnmarried
pre marry::aSpouseUnmarried
pre marry::differentGenders
post marry::isMarried
post marry::femaleHasMarriedHusband
post marry::maleHasMarriedWife
pre divorce::isMarried
pre divorce::isAlive
pre divorce::husbandAlive
pre divorce::wifeAlive
post divorce::isDivorced
post divorce::husbandDivorced
post divorce::wifeDivorced
pre death::isAlive
post death::notAlive
post death::husbandWidowed
post death::wifeWidowed
context Person::marry(aSpouse : Person)
pre differentGenders: (self.gender <> aSpouse.gender)
Sequence diagram
ada:Person
bob:Person
cyd:Person
dan:Person
birth('Ada',#female)
birth('Bob',#male)
marry(@bob)
birth('Cyd',#male)
divorce()
marry(@ada)
birth('Dan',#male)
death()
Class diagram
<<enumeration>>
Gender
female
male
Marriage
0..1
wife
husband
0..1
Person
name : String
civstat : CivilStatus
gender : Gender
alive : Boolean
birth(aName : String, aGender : Gender)
marry(aSpouse : Person)
divorce()
death()
spouse() : Person
<<enumeration>>
CivilStatus
single
married
divorced
widowed
Class extent
Person name civstat gender alive
ada 'Ada' #widowed #female true
bob 'Bob' #divorced #male true
cyd 'Cyd' #married #male false
dan 'Dan' #single #male true
Evaluate OCL expression
Enter OCL expression:
Person.allInstances->select(p|p.alive)->collect(p|Sequence{p.name,p.civstat})
Result:
Bag{Sequence{'Ada',#widowed},Sequence{'Bob',#divorced},Sequence{'Dan',#single}} : Bag(Sequence(OclAny))
Evaluate
Clear Result
Close
Ready.


context Person::marry(aSpouse : Person)
pre differentGenders: (self.gender <>
aSpouse.gender)


USE: c:/mg/civstat/scp-sub/civstat.use
File Edit State View Help
CivilStatusWorld
Classes
Person
Associations
Marriage
Invariants
Person::attributesDefined
Person::nameCapitalThenSmallLetters
Person::nameIsUnique
Person::femaleHasNoWife
Person::maleHasNoHusband
Pre-/Postconditions
pre birth::freshUnlinkedPerson
post birth::nameAssigned
post birth::civstatAssigned
post birth::genderAssigned
post birth::isAliveAssigned
pre marry::aSpouseDefined
pre marry::isAlive
pre marry::aSpouseAlive
pre marry::isUnmarried
pre marry::aSpouseUnmarried
pre marry::differentGenders
post marry::isMarried
post marry::femaleHasMarriedHusband
post marry::maleHasMarriedWife
pre divorce::isMarried
pre divorce::isAlive
pre divorce::husbandAlive
pre divorce::wifeAlive
post divorce::isDivorced
post divorce::husbandDivorced
post divorce::wifeDivorced
pre death::isAlive
post death::notAlive
post death::husbandWidowed
post death::wifeWidowed
context Person::marry(aSpouse : Person)
pre differentGenders: (self.gender <> aSpouse.gender)
Sequence diagram
ada:Person
bob:Person
cyd:Person
dan:Person
birth('Ada',#female)
birth('Bob',#male)
marry(@bob)
birth('Cyd',#male)
divorce()
marry(@ada)
birth('Dan',#male)
death()
Class diagram
<<enumeration>>
Gender
female
male
Marriage
0..1
wife
husband
0..1
Person
name : String
civstat : CivilStatus
gender : Gender
alive : Boolean
birth(aName : String, aGender : Gender)
marry(aSpouse : Person)
divorce()
death()
spouse() : Person
<<enumeration>>
CivilStatus
single
married
divorced
widowed
Class extent
Person name civstat gender alive
ada 'Ada' #widowed #female true
bob 'Bob' #divorced #male true
cyd 'Cyd' #married #male false
dan 'Dan' #single #male true
Evaluate OCL expression
Enter OCL expression:
Person.allInstances->select(p|p.alive)->collect(p|Sequence{p.name,p.civstat})
Result:
Bag{Sequence{'Ada',#widowed},Sequence{'Bob',#divorced},Sequence{'Dan',#single}} : Bag(Sequence(OclAny))
Evaluate
Clear Result
Close
Ready.