



Behavior Modeling State Diagrams

*Software Engineering and Databases Group
Department of Computer Languages and Systems
University of Seville
November 2015*

La traducción de este material docente ha sido financiada mediante la convocatoria 1.10B - Ayudas de innovación y mejora docente, convocatoria 2013-2014, modalidad B del II Plan Propio de Docencia de la Universidad de Sevilla. No ha habido financiación alguna para este proyecto de otros soportes.



1. Behavior model of a system
2. UML state diagrams
3. State diagrams and OCL
4. Alternative representations

Behavior Modeling: State Diagrams

• Learning objectives

- Understand the basics of behavior modeling using **UML state diagrams** (statecharts).
- Be able to **develop** state diagrams from a requirements specification.

Behavior Modeling: State Diagrams

Models of a software system

- **Static model (conceptual model)**
 - Describes the structure and constraints of the information representing the system **state**.
- **Behavior model**
 - Describes how the system **interacts** with actors and how its state evolves as a result of the interactions.

Model = {

$$\begin{aligned} \text{Static model} &= \left\{ \begin{array}{l} \text{Structure} \\ + \\ \text{Constraints} \end{array} \right. \\ + \\ \text{Behavior model} &= \left\{ \begin{array}{l} \text{External interactions} \\ + \\ \text{Internal evolution} \end{array} \right. \end{aligned}$$

Dicember 2014

Requirements Engineering

2

Behavior Modeling: State Diagrams

Behavior model of a software system

Behavior model

```

graph TD
    BM[Behavior model] --> EI[External interactions]
    BM --> IE[Internal evolution]
    EI --> UI[User Interface]
    EI --> SI[Service Interface]
    UI -.-> SO[System operations]
    SI -.-> SO
    SO -.-> SD[State diagrams]
    SD -.-> SO
  
```

User Interface
uses
Service Interface
System operations
must be consistent with
State diagrams

Dicember 2014

Requirements Engineering

3

Behavior Modeling: State Diagrams

- **State diagrams in UML (*statecharts*)**
 - If the objects of a class have a number of clearly identified **states**, a state diagram can be associated to that class.

Dicember 2014 Requirements Engineering 4

Behavior Modeling: State Diagrams

- **Elements of state diagrams**
 - **States:** represent **relevant** situations of the objects of a class in the problem domain.
 - They are usually named as **past participles**.

Dicember 2014 Requirements Engineering 5

Behavior Modeling: State Diagrams

Elements of state diagrams

- **Transitions:** indicate from which state to which state an object can pass when some **event** is triggered and/or some **condition** (guard) is met.

```

    graph TD
        start((Pseudoestado inicial)) -- "evento de creación 1" --> Estado1[Estado 1]
        start -- "evento de creación 2" --> Estado2[Estado 2]
        Estado1 -- "evento1" --> Estado3[Estado 3]
        Estado3 -- "evento 2 [condición 1]" --> Estado4[Estado 4]
        Estado3 -- "evento 2 [condición 2]" --> Estado5[Estado 5]
        Estado5 -- "evento 3" --> final1(((Final 1)))
        Estado5 -- "evento 3" --> final2(((Final 2)))
    
```

Dicember 2014 Requirements Engineering 6

Behavior Modeling: State Diagrams

Special states: initial pseudostate

- It is the only state in which object **creation events** are accepted. There can only be **one** in a diagram.
- Not an actual state (the object does not exist yet).

Dicember 2014 Requirements Engineering 7

Behavior Modeling: State Diagrams

Special states: final states

- They do not have output transitions, i.e. they are states in which the object **cannot evolve anymore**.
- There can be **several** of them in a state diagram.

```

    graph TD
        subgraph "stm Elementos diagrama estados"
            S1([Estado 1]) -- "evento de creación 1" --> SI((Pseudoestado inicial))
            S1 -- "evento de creación 2" --> S2([Estado 2])
            S1 -- "evento1" --> S3([Estado 3])
            S3 -- "evento 2 [condición 1]" --> S4([Estado 4])
            S3 -- "evento 2 [condición 2]" --> S5([Estado 5])
            S5 -- "evento 3" --> F1((Final 1))
            S5 -- "evento 3" --> F2((Final 2))
            F1 -- "Final 1" --> F1
            F2 -- "Final 2" --> F2
        end
    
```

Dicember 2014 Requirements Engineering 8

Behavior Modeling: State Diagrams

Special states: composite states

- They are states containing other internal state diagram.
- They can be represented collapsed or expanded.

```

    graph LR
        subgraph "stm Estado civil"
            V[Vivo]
            S1(( )) -- "nace" --> S1([Soltero])
            S1 -- "se casa" --> C([Casado])
            C -- "se casa" --> D([Divorciado])
            C -- "fallece cónyuge" --> V
            C -- "se separa" --> S2([Separado])
            S2 -- "vuelve con cónyuge" --> C
            S2 -- "se divorcia" --> D
            D -- "se casa" --> C
            D -- "se divorcia" --> S2
            S2 -- "muere" --> M((muerto))
        end
    
```

Dicember 2014 Requirements Engineering 9

Behavior Modeling: State Diagrams

Special states: concurrent states

- They are used when an object can be in **several states at the same time**.

stm Estados concurrentes

```

stateDiagram-v2
    [*] --> Soltero
    [*] --> Desempleado
    Soltero --> Casado : se casa
    Soltero --> Divorciado : se divorcia
    Casado --> Separado : se separa
    Casado --> Divorciado : se casa
    Separado --> Casado : vuelve con cónyuge
    Separado --> muerto : fallece cónyuge
    Divorciado --> Casado : se casa
    Divorciado --> Jubilado : se jubila
    Jubilado --> Empleado : lo contratan
    Jubilado --> Desempleado : se jubila
    Empleado --> Desempleado : lo despiden
    
```

The diagram illustrates concurrent states across two domains: Civil Status and Labor Status. In the Civil Status domain, it starts with an initial state (empty circle) leading to 'Soltero'. From 'Soltero', transitions to 'Casado' ('se casa') or 'Divorciado' ('se divorcia'). From 'Casado', transitions to 'Separado' ('se separa') or back to 'Casado' ('se casa'). From 'Separado', transitions to 'muerto' ('fallece cónyuge') or back to 'Casado' ('vuelve con cónyuge'). From 'Divorciado', transitions to 'Casado' ('se casa') or to 'Jubilado' ('se jubila'). In the Labor Status domain, it starts with an initial state (empty circle) leading to 'Desempleado'. From 'Desempleado', transitions to 'Empleado' ('lo contratan') or to 'Jubilado' ('se jubila'). From 'Empleado', transitions back to 'Desempleado' ('lo despiden') or to 'Jubilado' ('se jubila'). A final state 'muerto' is shown.

Dicember 2014 Requirements Engineering 10

Behavior Modeling: State Diagrams

Special states: concurrent states

- They are equivalent to a single state diagram with all possible state combinations.

stm Estados concurrentes

```

stateDiagram-v2
    [*] --> Soltero
    [*] --> Desempleado
    Soltero --> Casado : se casa
    Soltero --> Divorciado : se divorcia
    Casado --> Separado : se separa
    Casado --> Divorciado : se casa
    Separado --> Casado : vuelve con cónyuge
    Separado --> muerto : fallece cónyuge
    Divorciado --> Casado : se casa
    Divorciado --> Jubilado : se jubila
    Jubilado --> Empleado : lo contratan
    Jubilado --> Desempleado : se jubila
    Empleado --> Desempleado : lo despiden
    
```

This diagram is identical to the one above, illustrating concurrent states for Civil and Labor status using UML state diagrams.

Dicember 2014 Requirements Engineering 11

Behavior Modeling: State Diagrams

- Alternative representation: enumerations
 - The object state can also be represented by an **enumerated attribute**.

The diagram illustrates the equivalence between a statechart representation and an enumeration-based class definition. On the left, a statechart labeled 'stm C Statechart' shows a start node leading to state A, which then branches to states B and C. Both B and C transition to a final state. On the right, a UML class diagram labeled 'class Ejemplo statechart' shows a class 'C' with an attribute 'estado: EstadoC'. Below it is an enumeration named 'EstadoC' with values 'A', 'B', and 'C'.

Dicember 2014 Requirements Engineering 16

Behavior Modeling: State Diagrams

- Alternative representation: enumerations
 - Each **concurrent region** and each **composite state** need its own attribute and enumerated type.

The diagram shows a complex statechart with two regions: 'Estado civil' and 'Estado laboral'. The 'Estado civil' region contains states 'Vivo', 'Viudo', 'Casado', 'Separado', 'Soltero', and 'Divorciado', along with transitions like 'se casa', 'se separa', 'fallece cónyuge', 'vuelve con cónyuge', and 'nace'. The 'Estado laboral' region contains states 'Desempleado', 'Empleado', and 'Jubilado', with transitions like 'lo contratan', 'lo despiden', 'se jubila', and 'se jubila'. To the right, a UML class 'Persona' is defined with attributes 'estado: EstadoPersona', 'estadocivil: EstadoCivil', and 'estadolaboral: EstadoLaboral'. Below it are three enumerations: 'EstadoPersona' with values 'vivo' and 'muerto'; 'EstadoCivil' with values 'Soltero', 'Casado', 'Separado', 'Divorciado', and 'Viudo'; and 'EstadoLaboral' with values 'Desempleado', 'Empleado', and 'Jubilado'.

Dicember 2014 Requirements Engineering 17

Behavior Modeling: State Diagrams

UNIVERSIDAD DE SEVILLA
Escuela Técnica Superior de Ingeniería Informática
Departamento de Lenguajes y Sistemas Informáticos

1. Behavior model of a system
2. UML state diagrams
3. State diagrams and OCL
4. Alternative representations

Alternative representation: Booleans

- Each state can be represented by a **Boolean attribute**.
- The statechart structure is represented by **invariants**.

Dicember 2014 Requirements Engineering 18 © Universidad de Sevilla. Reservados todos los derechos. 2012

Behavior Modeling: State Diagrams

UNIVERSIDAD DE SEVILLA
Escuela Técnica Superior de Ingeniería Informática
Departamento de Lenguajes y Sistemas Informáticos

1. Behavior model of a system
2. UML state diagrams
3. State diagrams and OCL
4. Alternative representations

Bibliography

- C. Larman, ***UML y Patrones***. Ed. Prentice-Hall, 1999.
• Chapters 9 to 12
- C. Larman, ***UML y Patrones*** (2^a edición). Ed. Prentice-Hall, 2003.
• Chapters 10 to 12
- D. D'Souza y A. Wills, ***Objects, Components, and Frameworks with UML: The Catalysis Approach***. Ed. Addison-Wesley, 1999.
• Chapter 3

Dicember 2014 Requirements Engineering 19 © Universidad de Sevilla. Reservados todos los derechos. 2012

Behavior Modeling: State Diagrams

- **Comments, suggestions, ...**

U

S

E

V

I

L

T

A

R

D

M

C

S

P

H

O

N

G

F

E

V

A

R

D

M

C

S

P

H

O

N

G

E

V

A

R

D

M

C

S

P

H

O

N

G

E

V

A

R

D

M

C

S

P

H

O

N

G

E

V

A

R

D

M

C

S

P

H

O

N

G

E

V

A

R

D

M

C

S

P

H

O

N

G

E

V

A

R

D

M

C

S

P

H

O

N

G

E

V

A

R

D

M

C

S

P

H

O

N

G

E

V

A

R

D

M

C

S

P

H

O

N

G

E

V

A

R

D

M

C

S

P

H

O

<div style="position: absolute; top: 10px; left: 10px; width: 100px; height: 100px; border: 1px solid black; border-radius: 50%; background-color: white; display: