

Modelo Conceptual para Informação Espacial

Abordagem

Ponto de partida:

Modelo Entidade-Associação (EA)

intuitivo e largamente usado




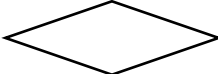
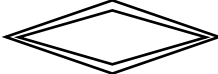
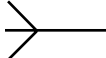
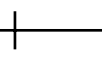
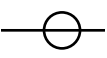
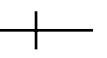
Objectivo:

Enriquecer o modelo EA com novas primitivas gráficas

semântica das novas primitivas descreve informação espacial

Diagrama e Modelo Entidade-Associação

- Um Diagrama EA é uma representação gráfica do Modelo EA
 - existem diversas notações gráficas
- O Modelo EA é caracterizado por 3 conceitos base:
 - Entidade, que tem existência independente dos restantes conceitos
 - Atributo, que caracteriza uma Entidade
 - Associação, que liga Entidades e tem multiplicidade e conectividade

Conceito	Símbolo Gráfico
Entidade e Entidade Fraca	 
Atributo	
Associação e Associação Fraca	 
Multiplicidade e Conectividade	   

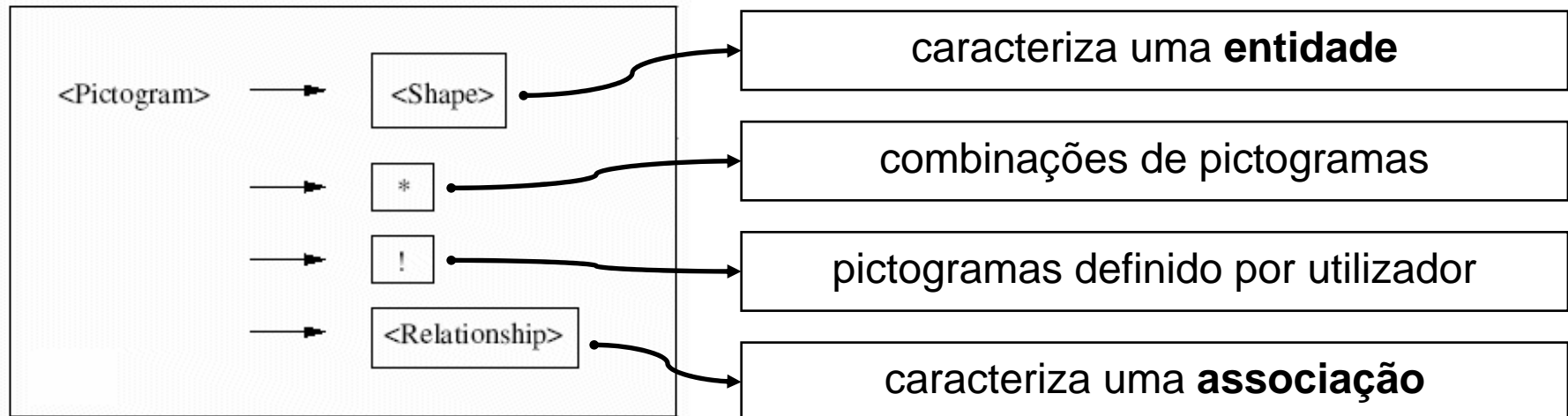
Estender o Modelo EA com Conceitos Espaciais

- Motivação
 - o EA baseia-se em conjuntos discretos sem relações implícitas
 - os dados espaciais vêm de conjuntos contínuos com relações implícitas
 - duas entidades espaciais têm sempre relações, e.g., distância, direcção
- ... desenhar explicitamente todas as relações espaciais
 - tornará demasiado confuso (ilegível) um diagrama EA
 - irá originar tabelas adicionais no esquema relacional
 - perde restrições implícitas nas relações espaciais (e.g., partições)
- Pictogramas – adicionados aos símbolos gráficos do EA permitem
 - etiquetar entidades espaciais e respectivos tipos de dados
 - realizar inferência sobre relações e restrições espaciais
 - simplifica a legibilidade do diagrama EA

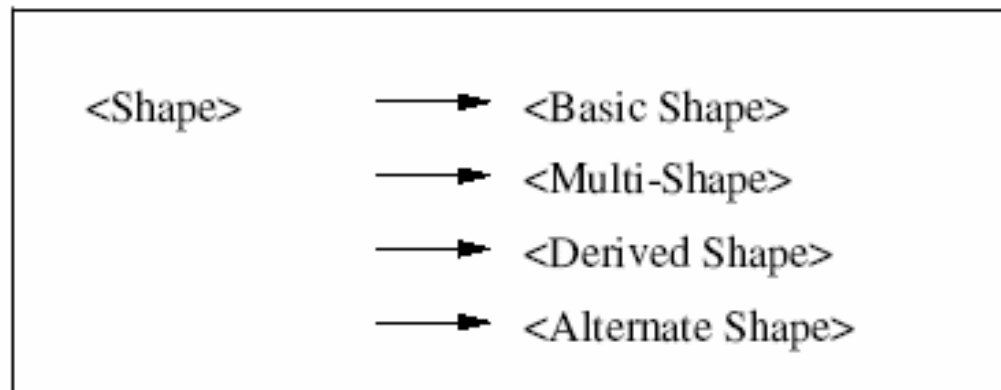
Pictograma – o que é?

Pictograma:

- Representação de um objecto inserido numa caixa gráfica.
 - Grafismo usado para estender o diagrama EA.
- Pode ter uma forma predefinida ou definida pelo utilizador.



Pictograma – *Shape*



Gramática do pictograma
do tipo *Shape* (Forma)

O arquétipo *Shape* constitui o elemento gráfico base do pictograma.

Representa os vários tipos geométricos no modelo espacial.

Pode ser: *basic-shape*, *multishape*, *derived-shape* ou *alternate-shape*

Pictograma – *Shape \ Basic-Shape*

<Shape>



<Basic Shape>



<Multi-Shape>



<Derived Shape>



<Alternate Shape>

Gramática do pictograma
do tipo *Shape* (Forma)

<BasicShape>



Gramática do pictograma
do tipo *Basic-Shape*

- *Basic-Shape* inclui os seguintes componentes-espaciais:

- ponto, linha e polígono



Point

Line

Polygon

- e.g., no Tema “Floresta” pode representar-se:

- torre de vigia como ponto (0-D)
- cada área florestal como um polígono (2-D)

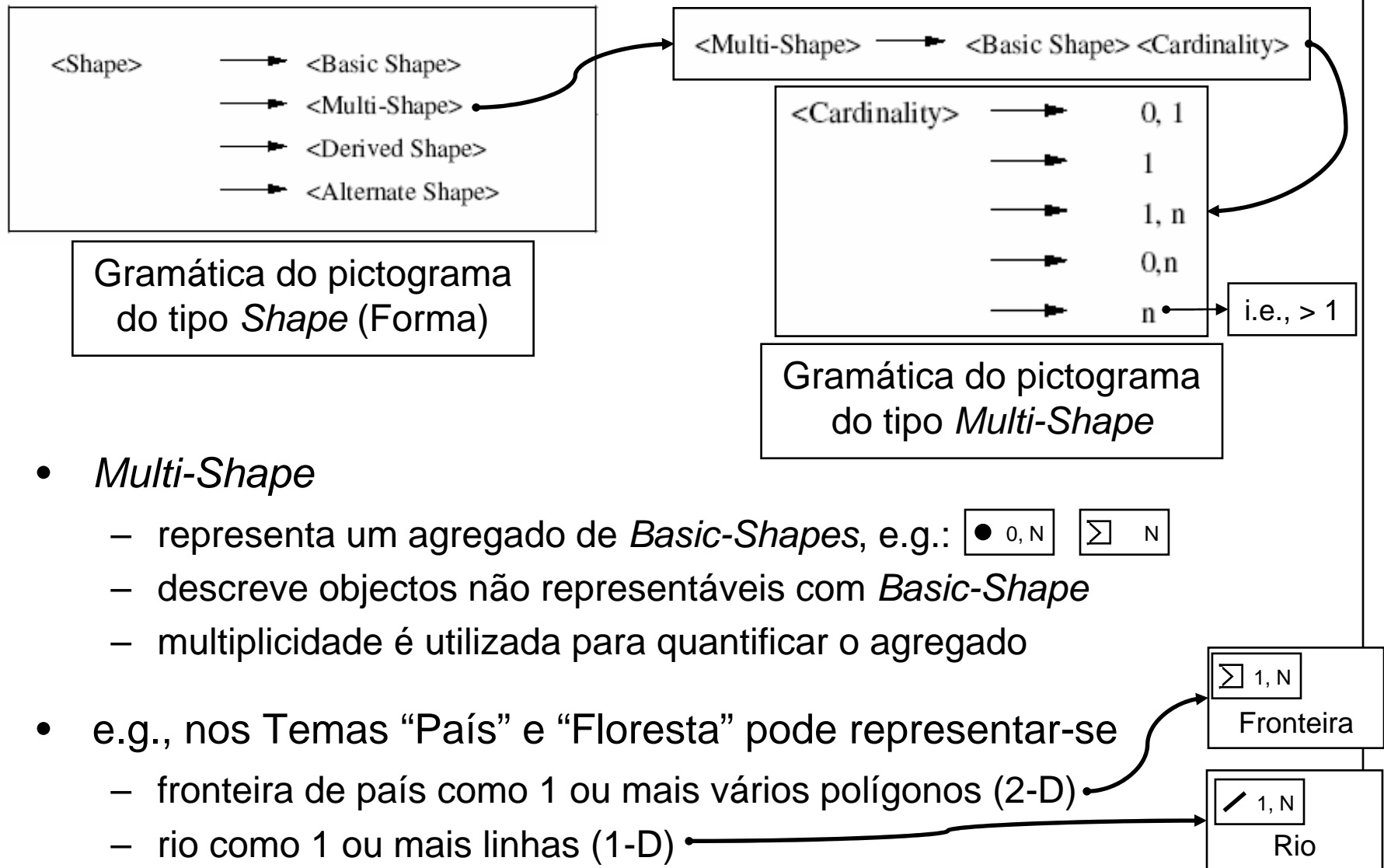


TorreVigia



AreaFlorestal

Pictograma – *Shape \ Multi-Shape*



Pictograma – *Derived-Shape*

<Shape>



<Basic Shape>



<Multi-Shape>



<Derived Shape>



<Alternate Shape>

Gramática do pictograma
do tipo *Shape* (Forma)

<Derived Shape>



<Basic Shape>

Gramática do pictograma
do tipo *Derived-Shape*

- *Derived-Shape*
 - representa um objecto que se constrói (é derivado) a partir de outros
- e.g., nos Temas “Cidade” e “Continente” pode representar-se
 - centro da cidade como um ponto calculado sobre o polígono (fronteira)
 - fronteira do Continente (e.g., Europa) a partir das fronteiras dos países



Pictograma – *Alternate-Shape*

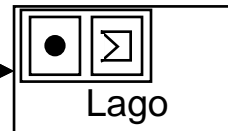
<Shape> → <Basic Shape>
 → <Multi-Shape>
 → <Derived Shape>
 → <Alternate Shape>

Gramática do pictograma
do tipo *Shape* (Forma)

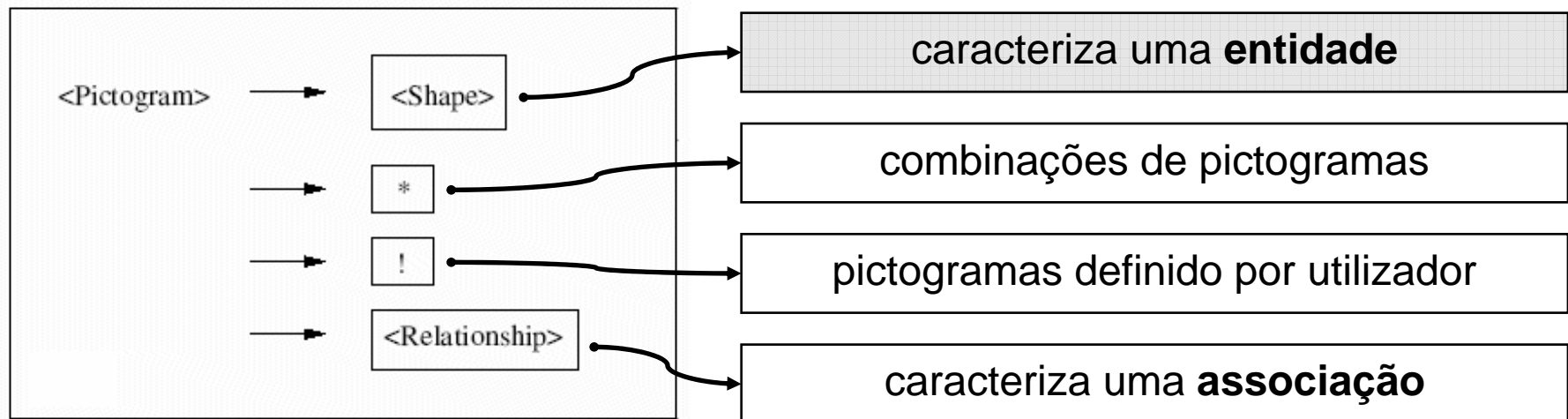
<Alternate Shape> → <Basic Shape> <Derived Shape>
 → <Basic Shape> <Basic Shape>

Gramática do pictograma
do tipo *Alternate-Shape*

- *Alternate-Shape*
 - representa objecto com forma (*Shape*) que depende de condição
- e.g., no Tema “Floresta” dependendo da escala considerada
 - um lago pode ser representar-se por um ponto (numa alta escala)
 - um lago pode ser representar-se por um polígono (numa baixa escala)

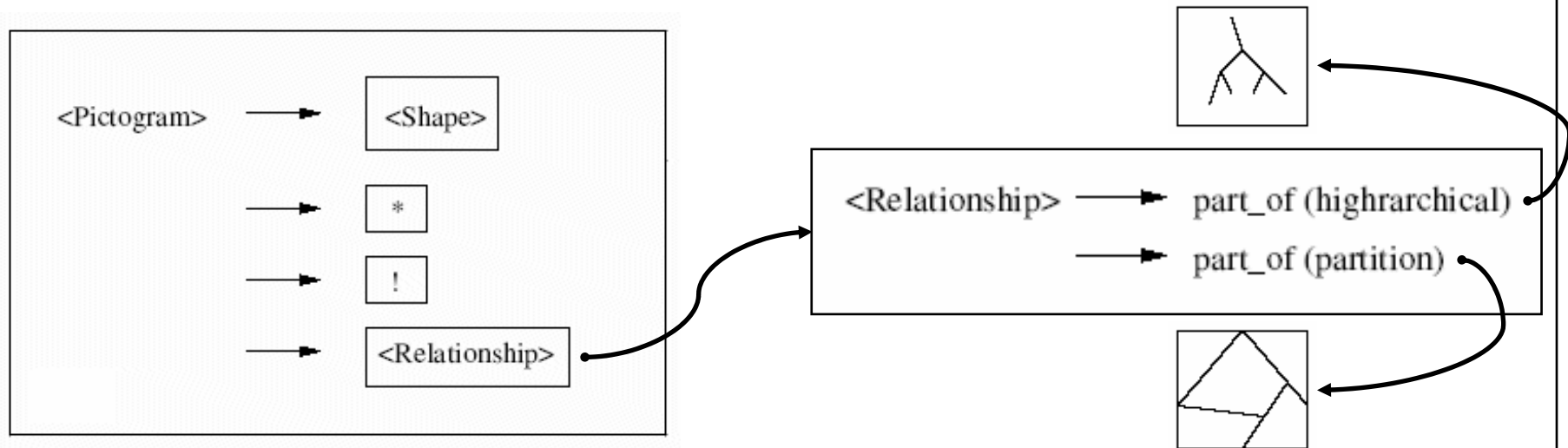


Pictograma – *Any-Shape* e *User-Shape*



- *Any-Shape* `*`
 - representa uma qualquer combinação de formas (*Shapes*)
 - i.e., qualquer geometria é admissível
- *User-Shape* `!`
 - representa um qualquer tipo de dados definido pelo utilizador
 - tem normalmente associada descrição textual sobre funcionalidade

Pictograma – *Relationship*



- *Relationship* é usado para modelar relações entre entidades
- *part_of (hierarchical)* – identificação “no todo” de “partes simples”
 - e.g., relação entre um grafo e um caminho nesse grafo
- *part_of (partition)* – separação “do todo” em “partes autónomas”
 - e.g., relação entre uma floresta e as suas partições em eco-sistemas



Um exemplo de domínio para modelar – “Parque Natural”

Um parque natural é constituído por várias florestas onde cada floresta consiste numa colecção de diversas zonas florestais cada uma com uma espécie predominante de árvores.

Cada floresta é acedida por estradas que conduzem ao centro de gestão (dessa floresta) e que daí percorrem o próprio parque.

No parque existem quartéis de bombeiros responsáveis por monitorar e controlar os incêndios que deflagram no parque. Para isso existem diversas torres de vigia espalhadas ao longo do parque.

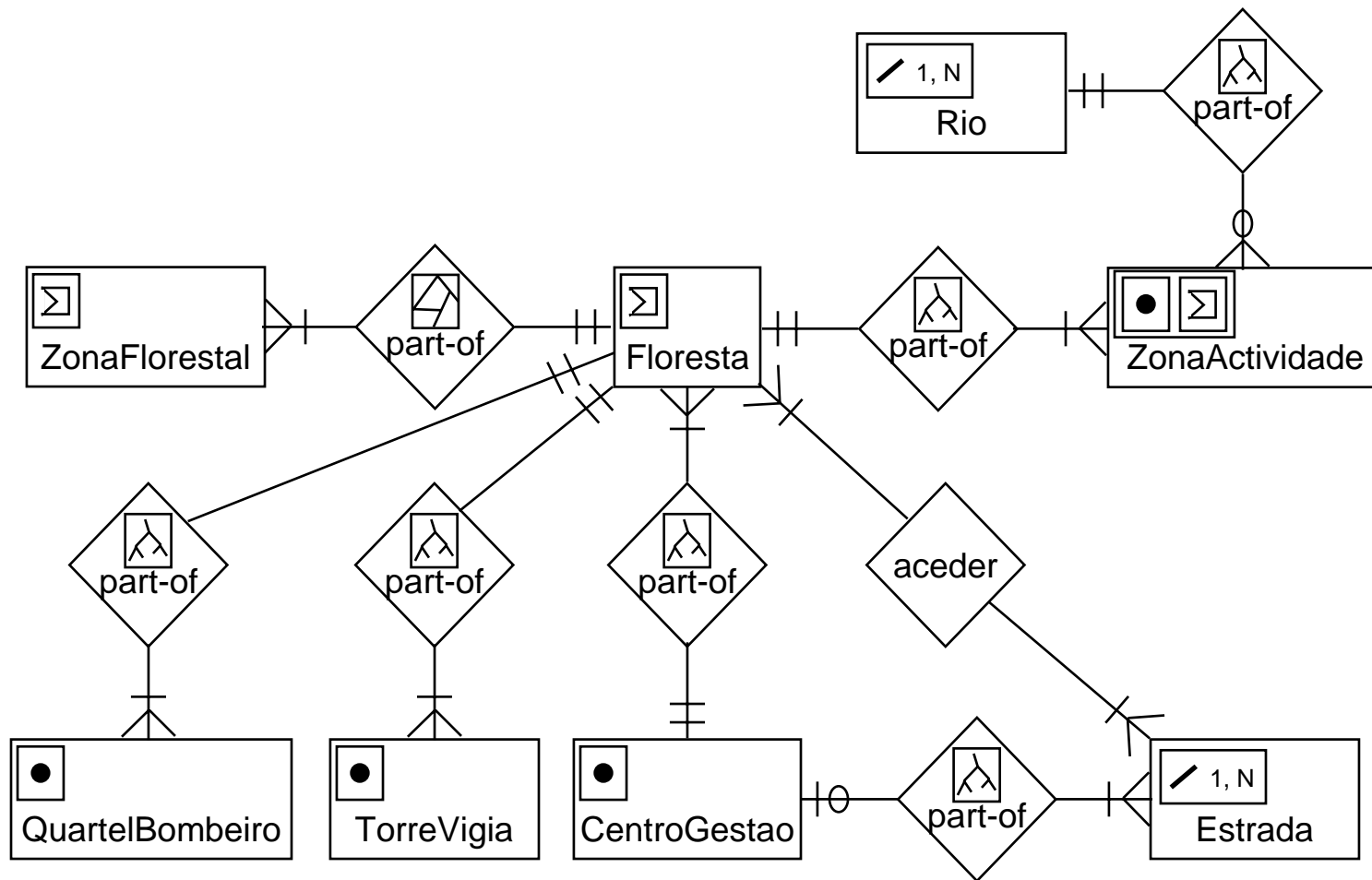
No parque existem pequenas zonas destinadas a determinadas actividades tais como acampamento por grupos de turistas e actividades de investigação por grupos de estudantes.

Finalmente, o parque é cruzado por vários rios e as zonas destinadas a actividades específicas estão sempre adjacentes a um rio.

Caracterização de Temas (no “Parque Natural”)

- Tema “Floresta”, que
 - inclui “Zonas Florestais”
 - inclui “Zonas de Actividades”
 - inclui “Quartéis de Bombeiros”
 - inclui “Torres de Vigias”
 - inclui “Centro de Gestão”
- Tema “Estrada” que
 - pode ser composto por uma ou mais “estradas”
- Tema “Rio” que
 - pode ser composto por um ou mais “rios”
 - inclui “Zona de Actividades” (na sua fronteira)

Modelo Entidade-Associação com Pictogramas



Tradução dos Pictogramas – Entidade

- Pictograma inserido numa Entidade
 - será traduzido no correspondente tipo de dados (SQL3)

Postgres Type	SQL92 or SQL3 Type	Description
bool	boolean	logical boolean (true/false)
box		rectangular box in 2D plane
char(n)	character(n)	fixed-length character string
cidr		IP version 4 network or host address
circle		circle in 2D plane
date	date	calendar date without time of day
float4/8	float(p)	floating-point number with precision p
float8	real, double precision	double-precision floating-point number
inet		IP version 4 network or host address
int2	smallint	signed two-byte integer
int4	int, integer	signed 4-byte integer
int4	decimal(p,s)	exact numeric for p <= 9, s = 0
int4	numeric(p,s)	exact numeric for p == 9, s = 0
int8		signed 8-byte integer
line		infinite line in 2D plane
lseg		line segment in 2D plane
money	decimal(9,2)	US-style currency
path		open and closed geometric path in 2D plane
point		geometric point in 2D plane
polygon		closed geometric path in 2D plane
serial		unique id for indexing and cross-reference
time	time	time of day
timespan	interval	general-use time span
timestamp	timestamp with time zone	date/time
varchar(n)	character varying(n)	variable-length character string

exemplo do
SGBD Posgres
(*open source*)

Tradução dos Pictogramas – Associação

- Pictograma inserido numa Associação
 - representa condições sobre a posição dos objectos (relações espaciais)
 - e será traduzido em relações de integridade espacial
- Relação de integridade espacial
 - podem ser suportadas pelo conceito de asserção (SQL3)
- Asserção (*assertion*)
 - consiste numa restrição inter-relação
 - é avaliada quando existe uma acção sobre a(s) relação(ões) relevante(s)

Validado ao nível do **esquema**:

```
CREATE ASSERTION <name>  
CHECK( <condition> )
```

≠

Validado ao nível da **tabela**:

```
CREATE TABLE <name> ...  
CHECK( <condition> )
```

tem valor true ou false

Exemplo – desenhar uma asserção

Considere-se o esquema:

$R1(\underline{k}, geo)$

$R2(\underline{k}, geo)$

onde `geo` representa um objecto-espacial do tipo polígono

Admita que qualquer objecto-espacial, `o1`, tem o método:

`overlap(o2)`, onde `o2` é um objecto-espacial, que devolve `true` se a intersecção entre `o1` e `o2` é não vazia, e `false` c.c.

Desenhar uma asserção para garantir que:

`T1` e `T2` são geometricamente disjuntos

Exemplo – desenhar uma asserção (cont.)

Considere-se o esquema:

R1(k, geo)

R2(k, geo)

onde geo representa um objecto-espacial do tipo polígono

Desenhar uma asserção para garantir que:

T1 e T2 são geometricamente disjuntos

```
CREATE ASSERTION a_geoDisjoint
CHECK
(NOT EXISTS
  (SELECT *
   FROM T1 as o1, T2 as o2
   WHERE o1.k <> o2.k AND
         o1.geo.overlap( o2.geo ) ) )
```

Asserção – usando *Postgres*

Actual versão do *Postgres* não suporta directamente a noção de “asserção”

Alternativa:

```
DROP FUNCTION f_geoDisjoint() CASCADE;  
CREATE FUNCTION f_geoDisjoint() RETURNS TRIGGER AS  
$$  
BEGIN  
IF  
    (EXISTS  
    (SELECT *  
     FROM T1 as o1, T2 as o2  
     WHERE o1.k <> o2.k AND  
           o1.geo.overlap( o2.geo ) ) )  
THEN  
    RAISE EXCEPTION 'disjoint constraint violation';  
END IF;  
$$  
LANGUAGE plpgsql;
```

```
CREATE TRIGGER trigger_geoDisjoint  
AFTER INSERT OR DELETE OR UPDATE ON T1  
FOR EACH ROW  
EXECUTE PROCEDURE f_geoDisjoint();
```

```
CREATE TRIGGER trigger_geoDisjoint  
AFTER INSERT OR DELETE OR UPDATE ON T2  
FOR EACH ROW  
EXECUTE PROCEDURE f_geoDisjoint();
```

Em síntese – alguns pictogramas propostos e analisados

