The \texttt{tkz-orm} package
Object-Role Model Drawing Library

Jakob Voß
jakob.voss@gbv.de

Version 0.1
January 25th, 2010

http://purl.org/net/tkz-orm

Abstract
This package provides styles for drawing Object-Role Model (ORM) diagrams in \TeX{} based on the PGF and TikZ picture environment.

Contents

1 Introduction \hfill 2

2 Object Types \hfill 3

3 Predicates and Roles \hfill 4

4 Constraints \hfill 7
  4.1 Uniqueness Constraints \hfill 7
  4.2 Mandatory Role Constraints \hfill 8
  4.3 External constraints \hfill 9
  4.4 Ring Constraints \hfill 11
  4.5 Number and Value Constraints \hfill 11
  4.6 Textual constraints \hfill 12

5 Additional Features \hfill 13
  5.1 Subtyping \hfill 13
  5.2 Duplicated and implied parts of a model \hfill 14
  5.3 Macros for text layout \hfill 16

6 Settings and Utilities \hfill 16

References \hfill 16

Index \hfill 17
1 Introduction

\texttt{tkz-orm} is intended to help you creating Object-Role Model (ORM) diagrams. It is based on the \texttt{pgf} and \texttt{Ti\textbackslash kZ}\textsuperscript{1} picture macro package for \LaTeX, and provides additional styles and commands to typeset ORM 2 diagrams.\textsuperscript{2} With \texttt{tkz-orm} you can “program” ORM diagrams just as you “program” your document when you use \LaTeX – including the inherent lack of \texttt{wysiwyg}. Unless multi-touch e-paper interfaces become usable, \texttt{tkz-orm} can best be combined with a whiteboard or paper and pencil — but you may also find ways to automatically create ORM diagrams with \texttt{tkz-orm}.

Status of this package

This is the first public version of \texttt{tkz-orm}. Please send your comments to the author so the package can be improved. All parts of the package are available at least under the \LaTeX\ Project Public License[\texttt{LPP08}] and the GNU Public license[\texttt{GPL91}]. For details have a look at the file \texttt{LICENSE} that is part of this package. The permanent URL of \texttt{tkz-orm} is \url{http://purl.org/net/tkz-orm} which redirects you to its current location.

ORM in a nutshell

Object-Role Modeling (ORM)\textsuperscript{3} is a fact-oriented modeling language that evolved from the Natural-language Information Analysis Method (NIAM) by G.M. Nijssen. The current version (ORM 2) is mainly based on works of Terry Halpin. Like ERM, UML, and other data modeling languages ORM helps to identify and abstract information objects, relationships, and rules in a Universe of Discourse to be formalized and implemented on another level. ORM includes a graphical notation and a defined verbalization in natural language. Models can further be validated by populating fact tables with sample data. An overview of the ORM 2 Graphical Notation is given in [Hal05] and more details in [HM08]. An ORM model consists of object types (section 2), \(n\)-ary predicates (section 3). Each of the \(n\) roles of an \(n\)-ary predicate is connected to an object type that plays the specific role in this predicate. Furthermore a model can contain constraints (section 4) and additional features (section 5). \texttt{tkz-orm} also allows you to change the appearance of ORM diagrams (section 6).

\begin{figure}[h]
\centering
\begin{tikzpicture}
\path[style=common] (0,0) node[]{A} (1,0) node[]{\{Value type\}} (2,0) node[]{Person with red hair} (3,0) node[]{Activity (Code)} (4,0) node[]{Type name (.reference)} (5,0) node[]{Zoomed};
\end{tikzpicture}
\caption{Examples of object types in ORM}
\end{figure}

\textsuperscript{1}\url{http://sourceforge.net/projects/pgf/}
\textsuperscript{2}This version is tested with \texttt{Ti\textbackslash kZ} version 2.00-cvs 2009-09-04. It will either be backported to \texttt{Ti\textbackslash kZ} version 2.00 or work with a yet to published new release of \texttt{Ti\textbackslash kZ}
\textsuperscript{3}\url{http://www.orm.net/}
2 Object Types

Object types are drawn as rectangles with rounded corners. The object’s type name is written as node text inside. Entity types use solid border lines and value types use dashed border lines. The minimal size of an object is set to 6mm×6mm. This package provides the following styles for entities and values:

/tikz/entity

This style is to be used with nodes that represent entity types.

/tikz/value

This style is to be used with nodes that represent value types.

/tikz/every entity
/tikz/every value=

Each of these styles is invoked by the styles entity or value. Change one of these styles to change the appearance of entity or value types.

/tikz/every object type

This style is invoked by the styles entity and value. Change this style to change the common appearance of entity and value types.

Since entity types and value types are very frequent node types in an ORM diagram, there are two special abbreviations for creating object types:

/entity

Inside \{tikzpicture\} this is an abbreviation for \node[entity].

/value

Inside \{tikzpicture\} this is an abbreviation for \node[value].
3 Predicates and Roles

The relationship parts (roles) played by objects are shown as boxes of fixed size (4mm×2.5mm). A predicate is a sequence of one or more concatenated role boxes. Predicates can be created with the following styles:

/tikz/roles=⟨number of roles⟩
/tikz/role= (alias for roles=1)
Shapes the current node as predicate with a given number of role boxes. Numbers from 1 to 20 are supported. The default value is 2 (binary).

A \begin{tikzpicture}[orm]
\entity (A) {A};
\entity (B) at (2.8,0) {B};
\entity (C) at (1.4,-1) {C};
\draw (A) to (B)
edge node [roles=3] (p) {} (A);
\draw (p.south) to (C);
\draw (A) to (C)
edge node [xshift=2mm] {} (C);
\end{tikzpicture}

/tikz/vroles=⟨number of roles⟩
/tikz/vrole= (alias for vroles=1)
Shapes the current node as predicate rotated by 90 degree (vertical).

/tikz/relation
/tikz/relationship
This styles are to be used with connection lines between objects and roles. By default it just sets the line width to 0.25mm by including the style every orm line. Alternatively of using this style you can enable the general style orm (see section 6) and just draw a normal line.

A \begin{tikzpicture}
\node[entity] (A) {A};
\node [roles] (r) at (1,0) {};
\draw [relation] (A) to (r);
\node [vrole] at (0,-.8) {} edge [relation] (A);
\end{tikzpicture}

Table 1 lists abbreviations that can be used for creating predicate nodes and relationship lines inside \{tikzpicture\}.

<table>
<thead>
<tr>
<th>Command(s)</th>
<th>Abbreviation for</th>
</tr>
</thead>
<tbody>
<tr>
<td>\unary or \role</td>
<td>\node[role]</td>
</tr>
<tr>
<td>\binary or \roles</td>
<td>\node[roles]</td>
</tr>
<tr>
<td>\ternary</td>
<td>\node[roles=3]</td>
</tr>
<tr>
<td>\vunary or \vrole</td>
<td>\node[vrole]</td>
</tr>
<tr>
<td>\vbinary or \vroles</td>
<td>\node[vroles]</td>
</tr>
<tr>
<td>\vternary</td>
<td>\node[vroles=3]</td>
</tr>
<tr>
<td>\plays</td>
<td>\draw[relationship]</td>
</tr>
</tbody>
</table>

Table 1: Abbreviations to create predicates, roles, and relationship lines

The general style of predicates and roles can be modified by the following keys:
By changing this style you can modify the common appearance of predicates.

```
\begin{tikzpicture}
\node (A) {A};
\entity (B) at (2,0) {B};
\entity (A) edge[relation] node[roles]{} (B);
\end{tikzpicture}
```

This style is evoked by the style `relationship`. To change the appearance of explicit relationship lines you can change this style. Please keep in mind that nodes placed on a line by `node` in one operation inherit properties from the line they refer to, so you should create relationship lines with `plays`.

```
\begin{tikzpicture}
\binary (r) at (1,0) {};
\plays (A) -- (r) -- (B);
\end{tikzpicture}
```

Predicates are drawn either horizontally (`roles`) or vertically (`vroles`) as nodes with one or more parts. Figure 3 shows some of the anchors. Vertical predicates are rotated by 90 degree so `north` is at the left, `west` is at the bottom etc.

```
\begin{tikzpicture}
\node[roles=3,shape example,inner ysep=0.75cm] (s) {};
\foreach \anchor/\placement in  
{one north/below, one south/above,  
  three north/below, three south/above,  
  one split/below, one split north/above, one split south/below,  
  two split/below, two split north/above, two split south/below,  
  north/below, south/above, east/below, west/above, center/above,  
  north west/above, north east/above, south west/below, south east/below}
{\draw[shift=(s.\anchor)] plot[mark=x] coordinates((0,0))
node[\placement] (s.\anchor);}
\end{tikzpicture}
```

Figure 2: Node anchors of an ORM predicate
The verbalization of a predicate can be given as \textbf{label} next to a predicate. For binary relationships forward and inverse readings can be separated by a slash.

To show the inverse reading, add an arrow tip with the commands \texttt{\ormleft} or \texttt{\ormup} (see section 5.3). Labels for predicates with more than two roles must contain three dots (\texttt{\ldots}) for each inner role. Role names and indices can be added by different styles.

\begin{tikzpicture}[orm]
\entity at (0,2) (P) {Person};
\entity at (3,2) (C) {Car};
\entity at (0,0) (L) {License};
\entity at (6.6,2) (M) {Maker};
\unary[label=valid] at (1.2,0) (V) {} edge (L);
\draw (P) to node[roles, label=below:drives,label=\ormleft{is driven by}]{} (C);
\draw (P) to node[vroles,label=has,label=below:\ormup{of}]{} (L);
\draw (C) to node[roles,label=created by/created]{} (M);
\ternary[label=below:drives \ldots by] at (3,1) (t) {};
\plays (P) -- (t.west);
\plays (C) -- (t);
\plays (M) -- (t.east);
\end{tikzpicture}

\texttt{/tikz/role name}

This style is to be used with role names. Role names can be displayed in square brackets and blue color next to a role box.

\begin{tikzpicture}[orm]
\entity (A) at (0,0) {A};
\entity (B) at (2.8,0) {B};
\plays (A) edge node[roles]{} (B);
\node[role name, at=(r.north), anchor=south east] {{[role1]}};
\node[role name, at=(r.north), anchor=south west] {{[role2]}};
\end{tikzpicture}

\texttt{/tikz/index=⟨n⟩:⟨index⟩}

Adds a role index as small label at the $n$th role box (default: $n$).
4 Constraints

ORM provides a rich set of constraints: Mandatory constraints (section 4.2) and uniqueness constraints (section 4.1) limit the way objects can be combined in predicates. External constraints (section 4.3) and subtype constraints (section 5.1) involve multiple roles or object types. All constraints are displayed in violet color and either drawn directly at an object type or role, or linked to one or more object types or role with dotted or dashed lines or arrows (see the styles limits and limits to). ORM 2 defines a set of symbols for external (section 4.3), ring (section 4.4) and other types of constraints. The general tkz-orm constraint key constraint only sets the font to violet. An optional key value can be used to add a predefined constraint symbol at the current position.

4.1 Uniqueness Constraints

By default every row in a fact table is unique. To express additional uniqueness constraints on one or more roles of a fact table or to explicitly express the uniqueness on the full predicate, a uniqueness bar is drawn above or below the fact roles. If the bar spans two or more non-adjacent roles, it is drawn as dotted line above or below the excluded roles. Bars can be stacked in multiple levels. To draw uniqueness bars you can use the following styles at predicate nodes:

/tikz/unique=⟨from⟩−⟨to⟩:⟨level⟩

Draws a uniqueness constraint bar above one or more roles. All parts of the key value are optional. As default a simple uniqueness bar above (⟨level⟩=1) the first role (⟨from⟩=1) is drawn. To make a bar span multiple roles, use the (⟨from⟩−⟨to⟩) syntax. Negative levels drawn the bar below the roles.

/tikz/skip unique=⟨from⟩−⟨to⟩:⟨level⟩

Draws a dotted uniqueness constraint bar. The syntax is the same as at the unique key. The bar includes background color in the gaps between dots, so it can be drawn on top of another bar.

/tikz/uniqueness bar
This style can be used to draw a line in the same style as a uniqueness constraint bar or a dotted uniqueness constraint bar.

```
\tikz\draw[uniqueness bar] (0,0) -- (2,0);
```

```
\tikz\draw[skipped uniqueness bar] (0,0) -- (2,0);
```

Please note that elementary $n$-ary predicates should only have uniqueness constraints of at least $n-1$ roles. Picture 3 shows how to split a ternary predicate with unique constraint bar on one role.

```
\begin{tikzpicture}[orm]
% needs positioning library
\ternary[unique] (t) at (0,0) {};
\entity[left=of t] {A} edge (t);
\entity[above=of t] {B} edge (t);
\entity[right=of t] {C} edge (t);
\node at (2.3,0) {$\Rightarrow$};
\entity (A) at (3,0) {A};
\binary[right=of A.north east,yshift=1mm,unique] (t1) {};
\binary[right=of A.south east,yshift=-1mm,unique] (t2) {};
\plays (A) -- (t1.west); \plays (A) -- (t2.west);
\entity[right=of t1] {B} edge (t1);
\entity[right=of t2] {C} edge (t2);
\end{tikzpicture}
```

Figure 3: A ternary predicate can be split into two binary predicates

4.2 Mandatory Role Constraints

To indicate explicitly that a role is mandatory, a mandatory role dot is added to either end of the line that connects the role to its object. Usually it is placed at the object type end. This package defines the style key `constraint dot` (alias `cdot`) and the following keys which can be used to add mandatory role dots to lines drawn with the `to` operation.

```
/tikz/constraint dot
/tikz/cdot
```

Draws the current node as mandatory role dot.

```
\tikz\node[cdot] {};
```

```
/tikz/mandatory
/tikz/required
```

This style enables the `relationship` style and add a mandatory role dot at the start of a straight line.
/tikz/required by

This style enables the relationship style and add a mandatory role dot at the end of a straight line.

/tikz/both required
/tikz/both mandatory

This style enables the relationship style and adds mandatory role dots at both ends of a straight line.

To show that either of many roles is mandatory, you can add an inclusive-or (disjunctive mandatory) role constraint with constraint=mandatory as shown in section 4.3. By default it is assumed that each entity or value must play at least some role. Independent object types whose roles are collectively optional can be marked by an exclamation mark appended to its name. It is recommended not to include implied mandatory constraints unless they refer to subtypes (section 5.1).

4.3 External constraints

External constraints span multiple roles that may come from different predicates. They are draw with circled symbol next to the roles, possibly linked to them with a dotted or dashed line (style limits and limits to).

/tikz/constraint=(constraint type)

This style sets the font to ORM style on constraint color (violet). If you provide a constraint type as key value, the current node is shaped as constraint circle and the symbol of the specified constraint type is drawn. The most common constraint types are exclusive (alias x) to indicate that populations of two or more role-sequences must be mutually exclusive, mandatory (alias required, total, and or) to indicate that each at least one of two more roles must be played by an object type, and xor (alias partition) to indicates that exactly one of two or more roles must be played by an object type. These constraints can also be used in subtyping (section 5.1). The constraint type unique and preferred unique enforces combinations
of object types that play a given set of roles to always be the same. The types \textit{equal}, \textit{subset}, and \textit{supset} indicate that tuples of roles have to be equal, subset or superset compared to each other (\textit{supset} is not included in standard ORM 2). The constraint type is \textit{misc} only draws the circle and can be used for custom constraints.

\begin{tikzpicture} [orm]
\matrix [column sep=2mm, row sep=2mm]
{ 
\node [constraint=x] {}; & \node [right] {exclusive / x}; & \node [constraint=or] {}; & \node [right, text width=2.8cm] {mandatory / total required / or}; & \node [constraint=xor] {}; & \node [right] {xor / partition}; \\
\node [constraint=unique] {}; & \node [right] {unique}; & \node [constraint=preferred unique] {}; & \node [right] {preferred unique}; & \node [constraint=misc] {}; & \node [right] {misc}; \\
\node [constraint=equal] {}; & \node [right] {equal}; & \node [constraint=subset] {}; & \node [right] {subset}; & \node [constraint=supset] {}; & \node [right] {supset}; 
};
\end{tikzpicture}

\textbf{/tikz/limits}  
This style is to be used with lines that connect constraint circles and roles. It can also be used to link other kinds of constraints (for instance value constraints) to the entity, value, or role they belong to.

\begin{tikzpicture} 
\unary (r1) at (0,0) {};
\unary (r2) at (0,-1.4) {};
\draw[limits] (r1) to node [pos=.4, name=s] {} (r2);
\node [constraint=subset] at (s) {};
\node [right=2mm of s, text width=2.3cm] {Each object that plays role 1 also plays role 2};
\end{tikzpicture}

\textbf{/tikz/limits to}  
This style is to be used with directed lines that connect constraint circles and roles. The line is drawn in the same style as \texttt{limits} but dashed and with an arrow tip of style \texttt{orm arrow} at the head.

\begin{tikzpicture} 
\unary (r1) at (0,0) {a};
\unary (r2) at (0,-1.4) {b};
\draw[limits to] (b) -- (a) node [pos=.4, name=s] {};
\end{tikzpicture}
\constraintdeclare \{⟨constraint type name⟩\}\{⟨path code⟩\}

This command declares a new constraint type. The \{⟨path code⟩\} is passed to the \texttt{append after} command key to be drawn after the constraint circle. Unless you want to extend ORM you do not need to declare new constraint types.

\constraintdeclarealias \{⟨alias name⟩\}\{⟨existing constraint type name⟩\}

This command can be used to create an alias (another name) for an existing constraint type.

/tikz/every constraint

This style is envoked at every constraint. You can change this style to change for instance the constraint color.

### 4.4 Ring Constraints

A ring constraint can be applied to any two roles of a predicate that are played by the same object type (or the same supertype). Such constraints can also be viewed as properties of a binary relation or as properties of a directed graph. This package provides the following symbols to depict ring constraints. The graphical syntax of some symbols is slightly changed compared to the official ORM 2 syntax and the two symbols for ‘asymmetric and intransitive’, and for ‘symmetric and intransitive’ have been omitted because of complexity.

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image" alt="irreflexive" /></td>
<td>irreflexive</td>
</tr>
<tr>
<td><img src="image" alt="asymmetric" /></td>
<td>asymmetric</td>
</tr>
<tr>
<td><img src="image" alt="intransitive" /></td>
<td>intransitive</td>
</tr>
<tr>
<td><img src="image" alt="antisymmetric" /></td>
<td>antisymmetric</td>
</tr>
<tr>
<td><img src="image" alt="acyclic" /></td>
<td>acyclic</td>
</tr>
<tr>
<td><img src="image" alt="acyclic intransitive" /></td>
<td>acyclic intransitive</td>
</tr>
<tr>
<td><img src="image" alt="symmetric" /></td>
<td>symmetric</td>
</tr>
<tr>
<td><img src="image" alt="purely reflexive" /></td>
<td>purely reflexive</td>
</tr>
<tr>
<td><img src="image" alt="symmetric irreflexive" /></td>
<td>symmetric irreflexive</td>
</tr>
</tbody>
</table>

\begin{tikzpicture}[orm]
\centering
\foreach \n in {0/irreflexive,1/asymmetric,2/intransitive,3/antisymmetric,4/acyclic,5/acyclic intransitive,6/symmetric,7/purely reflexive,8/symmetric irreflexive}{
\path ($mod(\n,3)*(3.4,0)-int(\n/3)*(0,0.8)$) node [constraint=\n] {};
}\end{tikzpicture}

\begin{tikzpicture}
\entity (P) {Person};
\binary[below=of P,unique=2,label=below:mother of] (r) {};
\plays (P) to (r.one north) (P) to (r.two north);
\limits (r.north) to +(4mm,0) node[constraint=acyclic]{};
\end{tikzpicture}

### 4.5 Number and Value Constraints

Value constraints, cardinality constraints, and occurrence frequencies can simply be drawn beside the object type or role they refer to, optionally linked to with a dotted or dashed limitation line.
**Frequency Constraints** specify the number of times an object can play a role. Usually it is connected to the roles with a limitation line.

\begin{tikzpicture}
\binary[index=1:1] (b) {};
\limits (b.one south) -- +(0,-.4) node[constraint]{f};
\end{tikzpicture}

A **Value constraint** indicates which values are allowed in an object type or role. It can be defined by declaring the set of possible values enclosed in curly brackets next to an object or role type. The command `\ormbraces` is a handy abbreviation to create curly brackets.

\begin{tikzpicture}
\entity (A) {A};
\node[constraint,anchor=north west,inner ysep=0] at (A.north east) \ormbraces{a, b, c};
\end{tikzpicture}

**Value comparision-constraints** are depicted by one of four comparision operators \(<\), \(\leq\), \(>\), and \(\geq\). The constraints are shown at a dashed arrow between two roles in the same way as `constraint=subset` and `constraint=supset` (but the value-comparision is between instances not between sets). Equality can be stated with `constraint=equal` which should not be confused with similar looking `constraint=purely reflexive`.

\begin{tikzpicture}
\foreach \x/\s in {0/<,1/ge,2/>,3/le}{
\draw[limits to] (\x,0) to (\x,1.2);
\node[constraint=\s] at (\x,.5) {};
}
\end{tikzpicture}

**Cardinality constraints** are rarely included in ORM diagrams since the are often implied by other constraints. However you can explicitely say that each population of an object type or a role includes exactly, at most, or at least a given number of instances. This is done by adding a cardinality constraint next to the object or role. The hash sign (“#”) stands for the cardinality.

\begin{tikzpicture}[orm]
\entity[label={[constraint]\#=$n$}] {A};
\entity[label={[constraint]below:1$\leq$\#$\leq$6}] at (1.1,0) {A};
\role[label={[constraint]\#$\leq$2}] at (2,0) {};
\end{tikzpicture}

### 4.6 Textual constraints

Constraints not expressed by predefined graphical notation may be specified as textual rules. Textual rules can be displayed as footnotes with footnote numbers or signs that mark the involved elements in the diagram.

`/tikz/rule=(mark)`

This key is to be used with nodes that contain textual rules. The optional \(\langle mark\rangle\) is shown as footnotes index left to the rule.

1 Each Number identifies at most one Room.
Each Number identifies at most one Room.

This command is an abbreviation for `\matrix[row sep=0mm,nodes={right}]` inside `{tikzpicture}`. Matrices are useful to draw multiple textual rules below each other.

For each Employee, birthdate < hiredate.

Each Employee who has Rank 'NonExec' uses at most one CompanyCar.

Each Employee who has Rank 'Exec' uses some CompanyCar.

5 Additional Features

5.1 Subtyping

To draw type hierarchies you can use the tree syntax of TikZ.
Draws a subtype relationships from the supertype to the subtype.

```
\begin{tikzpicture}
  \node[entity] (A) {Animal};
  \node[entity] (M) [right=6mm of A] {Mammal};
  \node[entity] (B) [below left=of M] {Bird};
  \node[entity] (P) [below left=of B] {Penguin};
  \node[entity] (P2) [below right=of B] {Parrot};
  \draw[subtype] (A) to (M);
  \draw[subtype] (A) to (B);
  \draw[subtype] (A) to (P);
  \draw[subtype] (A) to (P2);
\end{tikzpicture}
```

Works in the same way as `\texttt{subtype}` but with reverse direction.

```
\begin{tikzpicture}
  \node[entity] (A) {Animal};
  \node[entity] (S) [right=6mm of A] {Sheep};
  \draw[suptype] (S) to (A);
\end{tikzpicture}
```

Subtype constraints can be shown linked to the subtype arrows:

```
\begin{tikzpicture}[orm]
  \foreach \c/\x in {exclusive/0,total/2.5,partition/5}{
    \entity (A) at (\x,0) {A} [edge from parent/.style=subtype]
    \child {node[entity] (B) {B}} \child {node[entity] (C) {C}};
    \limits ($(A)!.7!(B)$) to node[constraint=\c] {} ($(A)!.7!(C)$);
    \node at (\x,-2) \c;
  }
\end{tikzpicture}
```

5.2 Duplicated and implied parts of a model

Sometimes an object type or predicate is referred to without describing all its details because it is defined in an external model or because it is shown duplicated at some other place in the same model. To indicate such an external or duplicated object type or a predicate, a shadow is added to its shape. Alternatively ORM 2 allows to add a circumflex “ˆ” to an object type’s name. A different kind of redundancy are roles and constraints that deduce from other
parts of the model. ORM 2 includes the possibility to shade redundant roles. This is useful for instance to show conceptual paths or join fact types that are normally excluded. Moreover ORM allows a zooming on object types. This means that only objects and roles connected to a given object type are shown.

/tikz/duplicated model

This style modifies the styles every orm object and every predicate so all object types and predicates in the current scope get a shadow.

\begin{tikzpicture}[orm]
\begin{scope}[duplicated model]
\entity (A) {A};
\node[role] (r1) [right=of A] {};
\node[role] (r2) [right=0 of r1] {};
\draw[relationship] (A) -- (r1);
\end{scope}
\entity (B) [right=of r2] {B};
\draw[orm] (r2) -- (B);
\end{tikzpicture}

/tikz/implied model

This style modifies the styles every orm line and every orm object in the current scope to draw all lines thin and all objects filled gray.

\begin{tikzpicture}
\matrix[column sep=4mm] {
\entity (A) {A}; & \binary (ab) {}; & \entity (B) {B}; & \binary (bc) {}; & \entity (C) {C}; \};
\plays (A) -- (ab) -- (B) -- (bc) -- (C);
\begin{scope}[implied model]
\node[constraint=unique] (con) [above=of B] {};
\limits (ab.one north) -- (con) -- (bc.two north);
\ternary[unique=1-3,skip unique=2] (abc) [below=4mm of B] {};
\plays (A) -- (abc.west); \plays (B) -- (abc); \plays (C) -- (abc.east);
\end{scope}
\end{tikzpicture}

/tikz/zoomed

This styles visualizes an object type as zoomed by using a thicker line.

\begin{tikzpicture}
\entity at (0,0) {Normal};
\entity[zoomed] at (2,0) {Zoomed};
\end{tikzpicture}
5.3 Macros for text layout

The following macros can be used both in Ti\textsc{kZ} pictures or normal text:

\texttt{\ormtext}

Sets the font to the same sans-serif variant which is used in ORM diagrams.

\texttt{\ormbf}

Sets the font to a bold variant of \texttt{\ormtext}.

\texttt{\ormc}

Sets the font to a \texttt{\ormtext} in constraint color.

\texttt{\ormsup{\langle text \rangle}}

Puts some text in a superscript variant of \texttt{\ormtext}.

\texttt{\ormsub{\langle text \rangle}}

Puts some text in a subscript variant of \texttt{\ormtext}.

\texttt{\ormind{\langle text \rangle}}

Puts some text in a superscript variant of \texttt{\ormbf}.

\texttt{\ormbraces{\langle text \rangle}}

Puts some text as \texttt{\ormtext} in braces.

\begin{quote}
1 A Person is not \{0,1\}, \textbf{Male} or \textbf{Female}, \up{up} or \left{left} but \textbf{queer} multi\textsup{multi}\textsup{gender}!
\end{quote}

6 Settings and Utilities

\texttt{/tikz/orm}

This style sets the font and line width and the default node distance

\texttt{/tikz/every orm line}

This style is envoked by all styles of this package that draw lines. By default it sets the line width to 0.3mm.

Changes

0.1, January 25th, 2010 First release
References


[LPP08] LaTeX Project Public License (LPPL) Version 1.3c, 2008.
Index

< constraint, 12
> constraint, 12
custom constraint, 10
equal constraint, 10
ge constraint, 12
le constraint, 12
mandatory constraint, 10
or constraint, 10
partition constraint, 10
preferred unique constraint, 10
required constraint, 10
subset constraint, 10
supset constraint, 10
total constraint, 10
unique constraint, 10
xor constraint, 10
x constraint, 10

Arrow tips
  mandatory, 10
\binary, 4
both mandatory key, 9
both required key, 9

cardinality constraints, 12
cdot key, 8
constraint key, 9
constraint dot key, 8
\constraintdeclare, 11
\constraintdeclarealias, 11
Constraints, 7
constraints
  cardinality, 12
textual, 12
  value-comparision, 12
values, 12
duplicated model key, 15

Entities, 3
\entity, 3
entity key, 3
every constraint key, 11
every entity key, 3
every object type key, 3
every orm line key, 16
every predicate key, 5
every relationship key, 5
every value key, 3

implied model key, 15
Independent object types, 9
index key, 6
\limits, 10
limits key, 10
limits to key, 10
mandatory key, 8
Object types, 3
  Independent, 9
orm key, 16
\ormbf, 16
\ormbraces, 16
\ormc, 16
\ormind, 16
\ormsub, 16
\ormsup, 16
\ormtext, 16
\plays, 4
predicates, 4
relation key, 4
relationship key, 4
required key, 8
required by key, 9
\role, 4
role key, 4
role name key, 6
roles, 4
\roles, 4
roles key, 4
rule key, 12
\rules, 13
skip unique key, 7
skipped uniqueness bar key, 8
subtype key, 14
supertype key, 14
\ternary, 4
textual constraints, 12
\unary, 4

18
unique key, 7
uniqueness bar key, 7
\value, 3
value key, 3
value constraints, 12
value-comparison constraints, 12
Values, 3
\vbinary, 4
\vrole, 4
vrole key, 4
\vroles, 4
vroles key, 4
\vternary, 4
\vunary, 4
zoomed key, 15