## Assessing bidirectional model transformation tools

Alcino Cunha


Universidade do Minho

BIRS BX workshop, Banff
December, 2013

## Motivation

- Survey (operational) BX tools
- Focus on expressiveness
- Example driven
- Systematic exploration of the consistency design space
- Clarification of supported BX laws
- Repository of unit tests for (future) tool developers


## Constraint maintainers

$$
\begin{aligned}
& \mathrm{R} \subseteq A \times B \\
& \overrightarrow{\mathrm{R}}: A \times B \hookrightarrow B \\
& \stackrel{\rightharpoonup}{\mathrm{R}}: A \times B \hookrightarrow A
\end{aligned}
$$

## Basic properties of relations

$$
\begin{equation*}
\frac{\mathrm{R}(a, b) \quad \mathrm{R}\left(a, b^{\prime}\right)}{b=b^{\prime}} \quad \mathrm{R} \subseteq A \times \text { lone } B \tag{SIMPLE}
\end{equation*}
$$

$$
\frac{\mathrm{R}(a, b) \mathrm{R}\left(a^{\prime}, b\right)}{a=a^{\prime}} \quad \mathrm{R} \subseteq A \text { lone } \times B
$$

$$
\overline{\exists b . R(a, b)} \quad \mathrm{R} \subseteq A \times \text { some } B
$$

(Entire)

$$
\overline{\exists a . R(a, b)} \quad \mathrm{R} \subseteq A \text { some } \times B
$$

(Surjective)

## The bestiary of relations



## Data domain

- Shape
- Tree
- Graph
- Constraints
- None
- Order
- Keys
- Other


## Some laws of interest

$$
\frac{\overleftarrow{\mathrm{R}}\left(a, b^{\prime}\right)=a^{\prime}}{\mathrm{R}\left(a^{\prime}, b^{\prime}\right)}
$$

(Correctiness)

$$
\begin{equation*}
\frac{\mathrm{R}\left(a, b^{\prime}\right)}{\overline{\mathrm{R}}\left(a, b^{\prime}\right)=a} \tag{Stabllity}
\end{equation*}
$$

$$
\frac{\exists a^{\prime} \cdot R\left(a^{\prime}, b^{\prime}\right)}{\widehat{R}\left(a, b^{\prime}\right) \downarrow}
$$

(Totality)

$$
\frac{\overleftarrow{R}\left(a, b^{\prime}\right)=a^{\prime} \quad R\left(a^{\prime}, b\right)}{\overleftarrow{R}(a, b)=a^{\prime}}
$$

## For curators

- Propose simple but illustrative examples:
- Specify meta-models + consistency relation + unit tests.
- UML diagrams + (hopefully) non-ambiguous natural language?
- Propose and formalize laws to be evaluated.
- Synthesize results by example (tool vs law).
- Synthesize global results (tool vs consistency vs domain).
- Publish results in a web-site (BX Wiki?).


## For tool developers

- For each example:
- Show how meta-models + consistency relation can be encoded.
- Describe how unit tests can be implemented:
- Show how to encode models.
- Describe tool configuration.
- Report results.
- Ideally, package tool + examples in a virtual machine.


## Bijection \#1 (one $\rightarrow$ one), Sets

| Person |
| :---: |
| name : String |


| Employee |
| :---: |
| name : String |

Every person is an employee and vice-versa

| : Person |
| :---: |
| name $=$ " $\mathrm{A} "$ |


$\varnothing$

| :Person | :Person | :Employee |
| :---: | :---: | :---: |
| name $=$ " A " | name = "A" | name $=$ " ${ }^{\text {A }}$ |


| :Person | :Person | :Employee | :Employee |
| :---: | :---: | :---: | :---: |
| name = "A" | name $=$ " B " | name = "B" | name $=$ " ${ }^{\text {B" }}$ |

## Bijection \#2 (one $\rightarrow$ one), Sets with keys

| Person |
| :---: |
| name : String \{ id \} |


| Employee |
| :---: |
| name: String $\{$ id $\}$ |

Every person is an employee and vice-versa


| :Person |
| :---: |
| name $=$ " A " |


| :Employee |
| :---: |
| name $=$ " $\mathrm{B} "$ |


| :Person | :Person |
| :---: | :---: |
| name $=$ " $\mathrm{A} "$ | :Employee :Employee <br> name $=$ " $\mathrm{B} "$  $\mathrm{name}=$ " $\mathrm{B} "$ |

## Surjection \#1 (some $\rightarrow$ one), Sets with keys

| Person |
| :--- |
| name : String $\{$ id $\}$ <br> age : Integer |


| Employee |
| :---: |
| name $:$ String $\{$ id $\}$ |

Every person is an employee and vice-versa
$\varnothing$

| :Person |
| :--- |
| name $=$ "A" <br> age $=30$ |


| :Person |
| :--- |
| name $=$ "A" |
| age $=30$ |


| :Person |
| :--- |
| name $=$ " $\mathrm{B} "$ |
| age $=40$ |


| :Employee |
| :---: |
| name $=$ "A" |


| $:$ Person | $:$ Person |
| :--- | :--- |
| name $=" \mathrm{~A} "$ <br> age $=30$ | name $=$ " $\mathrm{B} "$ <br> age $=40$ |


| :Employee |
| :---: |
| name $=$ " B " |


| :Employee |
| :---: |
| name $=$ "C" |

## Surjection \#2 (some $\rightarrow$ one), Sets with order

| Person |
| :---: |
| names : String [1..*] \{ ordered \} |


| Employee |
| :---: |
| name : String |

Every person is an employee with its first name and vice-versa

| :Person |
| :---: |
| names : $\left\{{ }^{\prime} \mathrm{A}^{\prime}\right.$, " A ", " B " $\}$ |


| :Person |
| :---: |
| names : $\{$ " A ", " A ", " B " $\}$ |



| :Employee |
| :---: |
| name $=$ " $"$ " |


| :Person |
| :---: |
| names : $\{$ "A", "A", "B" $\}$ |



| :Employee |
| :---: |
| name $=$ "A" |


| :Employee |
| :---: |
| name $=$ " $\mathrm{B} "$ |

## Injection \#1 (lone $\rightarrow$ one), Sets with keys

| Person |
| :---: |
| name : String \{ id \} |


| Employee |
| :---: |
| name : String |

Every person is an employee and vice-versa


| $\begin{array}{\|r} \hline \text { :Person } \\ \hline \text { name }=\text { "A" } \end{array}$ | $\begin{array}{\|r\|} \hline \text { :Person } \\ \hline \text { name = "B" } \\ \hline \end{array}$ | $\begin{gathered} \text { :Employee } \\ \hline \text { name }=\text { "A" } \\ \hline \end{gathered}$ |  |
| :---: | :---: | :---: | :---: |
| :Person | :Person | :Employee | :Employee |
| name = "A" | name $=$ " B " | name $=$ " ${ }^{\text {B }}$ | name $=$ " ${ }^{\text {B }}$ |

## Function \#1 (set $\rightarrow$ one), Sets with keys

| Person |
| :--- |
| name : String $\{$ id $\}$ <br> age : Integer |


| Employee |
| :---: |
| name : String |

Every person is an employee and vice-versa
$\varnothing$

| :Person |
| :--- |
| name $=$ "A" <br> age $=30$ |


| :Person |
| :--- |
| name $=$ "A" |
| age $=30$ |


| :Person |
| :--- |
| name $=$ " $\mathrm{B} "$ |
| age $=40$ |


| :Employee |
| :---: |
| name $=$ "A" |


| $:$ Person | $:$ Person |
| :--- | :--- |
| name $=" \mathrm{~A} "$ <br> age $=30$ | name $=$ " $\mathrm{B} "$ <br> age $=40$ |


| :Employee |
| :---: |
| name $=$ " B " |


| :Employee |
| :---: |
| name $=$ " $\mathrm{B} "$ |

## Surjective entire relation \#1 (some $\times$ some), Sets

| Person |
| :---: |
| name: String |


| Employee |
| :---: |
| name: String |

Any pair of models is consistent


| :Person |
| :---: |
| name $=$ "A" |


| :Person |
| :---: |
| name $=$ "B" |

## Surjective entire relation \#2 (some $\times$ some), Sets

| Person |
| :---: |
| name: String |


| Employee |
| :---: |
| name: String |

There are persons iff there are employees

:Person
name = "A"
$\varnothing$


| :Person | :Person |
| :--- | :--- |
| name $=$ "A" | :Per$\quad$ name $=\mathrm{B} "$ |

:Employee
name = "C"

## We want you!



## Acknowledgements

- Perdita Stevens (inspiration, examples)
- Soichiro Hidaka (GRoundTram)
- Nuno Macedo (Echo)
- Tiago Jorge (web-site)

