Abstract

Described are computer-based methods and apparatuses, including computer program products, for declarative and unified data transition. In some embodiments, a computer implemented method includes determining an unified configuration for a knowledge domain. The unified configuration can include one or more predicates for one or more system objects, and/or one or more relationships between the one or more system objects. The method can include generating one or more transformation rules based on the one or more predicates. The one or more transformations can enable transformation of input data. The input data can include information associated with the one or more system objects and/or the one or more relationships between the one or more system objects. The method can include generating one or more reconciliation rules based on the one or more predicates. The one or more reconciliation rules can be associated with a destination database and can enable reconciliation of the transformed data with the destination database.
FIG. 1

Mediation Transition System 110

Operational Support Database 123

Operational Support System 120

External System Database A 140a

External System Database B 140b

External System Database C 140c

...
### BSCBoard

<table>
<thead>
<tr>
<th>FDN</th>
<th>Serial Number</th>
<th>Device</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>123</td>
<td>123456</td>
</tr>
<tr>
<td>2</td>
<td>321</td>
<td>NULL</td>
</tr>
<tr>
<td>3</td>
<td>Test</td>
<td>456879</td>
</tr>
<tr>
<td>4</td>
<td>789</td>
<td>123000</td>
</tr>
</tbody>
</table>

### BSCFrame

<table>
<thead>
<tr>
<th>FDN</th>
<th>Serial Number</th>
<th>Device</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>000</td>
<td>888</td>
</tr>
<tr>
<td>2</td>
<td>Test1</td>
<td>1,22</td>
</tr>
<tr>
<td>3</td>
<td>(\frac{1}{2})</td>
<td>Null</td>
</tr>
<tr>
<td>4</td>
<td>951</td>
<td>753654</td>
</tr>
</tbody>
</table>

### BSCDevice

<table>
<thead>
<tr>
<th>Device</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
</tr>
<tr>
<td>2</td>
</tr>
<tr>
<td>3</td>
</tr>
<tr>
<td>4</td>
</tr>
<tr>
<td>5</td>
</tr>
</tbody>
</table>

---

**FIG. 6A**
Unified Configuration

<configuration name="MyConfiguration"
  xmlns="http://www.netcracker.com/schemas/mediation/transition">
  <entity-set name="SDB">
    <entity name="BSCBoard">
      <attribute name="FDN"/>
      <attribute name="Serial_Number"/>
      <attribute name="Device" relation="Card to Device"/>
    </entity>
    <entity name="BSCFrame">
      <attribute name="FDN"/>
      <attribute name="Serial_Number"/>
      <attribute name="Device" relation="Card to Device"/>
    </entity>
    <entity name="BSCDevice">
      <attribute name="Device"/>
    </entity>
    <relationship name="Board to Device" multiplicity="0..1" direction="in">
      <source-entity>BSCBoard</source-entity>
      <destination-entity>Device</destination-entity>
    </relationship>
    <relationship name="Frame to Device" multiplicity="0..1" direction="in">
      <source-entity>BSCFrame</source-entity>
      <destination-entity>Device</destination-entity>
    </relationship>
  </entity-set>
  <entity-set name="IDB Entity Set">
    <entity name="Card" pk="Card ID">
      <attribute name="Card ID" type="integer"/>
      <attribute name="Parent ID" type="integer" relation="Card to Device"/>
      <attribute name="Name" type="string"/>
    </entity>
    <entity name="Device" pk="Device ID">
      <attribute name="Device ID" type="integer" relation="Card to Device"/>
    </entity>
    <relationship name="Card to Device" multiplicity="0..1" direction="in">
      <source-entity>Card</source-entity>
      <destination-entity>Device</destination-entity>
    </relationship>
  </entity-set>
</configuration>

FIG. 6B
<configuration name="MyConfiguration">
  xmlns="http://www.jeroenjager.com/schemas/mediation/transition"
  xmlns:v="http://www.netcracker.com/schemas/mediation/transition/validation">
  <entity-set name="SDB">
    <entity name="BSCBoard">
      <attribute name="FDN"/>
      <attribute name="Serial_Number">
        <v:is-not-null name="IsNotNull 1 Validation Rule"
        majority="major"/>
      </attribute>
    </entity>
    <entity name="BSCFrame">
      <attribute name="FDN"/>
      <attribute name="Serial_Number">
        <v:is-not-null name="IsNotNull 2 Validation Rule"
        majority="major"/>
      </attribute>
    </entity>
    <entity name="BSCDevice">
      <attribute name="Device">
        <v:like name="Like Validation Rule" majority="major">
          <v:expression>Dev%</v:expression>
        </v:like>
      </attribute>
    </entity>
    <relationship name="Board to Device" multiplicity="0..1" direction="in">
      <source-entity>BSCBoard</source-entity>
      <destination-entity>Device</destination-entity>
    </relationship>
    <relationship name="Frame to Device" multiplicity="0..1" direction="in">
      <source-entity>BSCFrame</source-entity>
      <destination-entity>Device</destination-entity>
    </relationship>
  </entity-set>
</configuration>

FIG. 6C
Validated Data

<table>
<thead>
<tr>
<th>FDN</th>
<th>Serial Number</th>
<th>Device</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>123</td>
<td>123456</td>
<td>Dev_1</td>
<td></td>
</tr>
<tr>
<td>321</td>
<td>NULL</td>
<td>1231</td>
<td></td>
</tr>
<tr>
<td>Test</td>
<td>456879</td>
<td>Dev_3</td>
<td></td>
</tr>
<tr>
<td>7891</td>
<td>23000</td>
<td>Dev_2</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>FDN</th>
<th>Serial Number</th>
<th>Device</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>000</td>
<td>888</td>
<td>Dev_1</td>
<td></td>
</tr>
<tr>
<td>Test</td>
<td>11,22</td>
<td>Dev_3</td>
<td></td>
</tr>
<tr>
<td>½</td>
<td>Null</td>
<td>1231</td>
<td></td>
</tr>
<tr>
<td>951</td>
<td>753654</td>
<td>Dev_2</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Device</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Dev_1</td>
<td></td>
</tr>
<tr>
<td>Dev_2</td>
<td></td>
</tr>
<tr>
<td>Dev_3</td>
<td></td>
</tr>
<tr>
<td>Xyz_3</td>
<td></td>
</tr>
<tr>
<td>1231</td>
<td></td>
</tr>
</tbody>
</table>

Deleted according to "IsNotNull 1 Validation Rule"

Deleted according to "IsNotNull 2 Validation Rule"

Deleted according to "Like Validation Rule"

Deleted according to "Like Validation Rule"

FIG. 6D
<configuration name="MyConfiguration"
  xmlns="http://www.netcracker.com/schemas/mediation/transition"
  xmlns:tr="http://www.netcracker.com/schemas/mediation/transition/transformation">
  <entity-set name="IDB Entity Set">
    <entity name="Card" pk="Card ID">
      <attribute name="Card ID">
        <tr:expression transformation="Card Transformation">
          <tr:source-attribute entity="BSCBoard" name="FDN"/>
        </tr:expression>
      </attribute>
      <attribute name="Parent ID" relation="Card to Device">
        <tr:expression transformation="Card Transformation">
          <tr:source-attribute entity="BSCBoard" name="Device"/>
        </tr:expression>
      </attribute>
      <attribute name="Name" type="string">
        <tr:expression transformation="Card Transformation" merge-
        rule="F_SN||"||B_SN">
          <tr:source-attribute entity="BSCFrame" name="Serial Number"
            alias="F_SN"/>
          <tr:source-attribute entity="BSCBoard" name="Serial Number"
            alias="B_SN"/>
        </tr:expression>
      </attribute>
    </entity>
    <entity name="Device" pk="Device ID">
      <attribute name="Device ID" relation="Card to Device">
        <tr:expression transformation="Device Transformation">
          <tr:source-attribute entity="Device" name="Device"/>
        </tr:expression>
      </attribute>
      <attribute name="Device">
        <tr:expression transformation="Device Transformation">
          <tr:source-entity name="Device" alias="DEV"/>
        </tr:expression>
      </attribute>
    </entity>
    <relationship name="Card to Device" multiplicity="0..1" direction="in">
      <source-entity>Card</source-entity>
      <destination-entity>Device</destination-entity>
    </relationship>
  </entity-set>
</configuration>

FIG. 6E
### Card

<table>
<thead>
<tr>
<th>Card ID</th>
<th>Parent ID</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Dev_1</td>
<td>123456-888</td>
</tr>
<tr>
<td>2</td>
<td>Dev_3</td>
<td>456879-1,22</td>
</tr>
<tr>
<td>3</td>
<td>Dev_2</td>
<td>123000-753654</td>
</tr>
</tbody>
</table>

### Device

<table>
<thead>
<tr>
<th>Device ID</th>
<th>Device Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Dev_1</td>
</tr>
<tr>
<td>2</td>
<td>Dev_2</td>
</tr>
<tr>
<td>3</td>
<td>Dev_3</td>
</tr>
</tbody>
</table>
Post-Transformation Validation Rules

600g

<entity-set name="IDB Entity Set">
  <entity name="Card" pk="Card ID">
    <attribute name="Card ID"/>
    <attribute name="Parent ID" relation="Card to Device"/>
    <attribute name="Name" type="string"/>
  </entity>
  <entity name="Device" pk="Device ID">
    <attribute name="Device ID" type="integer" relation="Card to Device"/>
  </entity>
  <relationship name="Card to Device" multiplicity="0..1" direction="in">
    <source-entity>Card</source-entity>
    <destination-entity>Device</destination-entity>
  </relationship>
</entity-set>
Validated Transformed Data

<table>
<thead>
<tr>
<th>Card ID</th>
<th>Parent ID</th>
<th>Name</th>
<th>Device ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>123</td>
<td>Dev_1</td>
<td>123456-888</td>
<td>Dev_1</td>
</tr>
<tr>
<td>789</td>
<td>Dev_3</td>
<td>456879-1.22</td>
<td>Dev_3</td>
</tr>
<tr>
<td></td>
<td>Dev_2</td>
<td>123000-753654</td>
<td></td>
</tr>
</tbody>
</table>

FIG. 6H
<configuration name="MyConfiguration">
  <entity-set name="IDB Entity Set">
    <entity name="Card" pk="Card ID">
      <recobject-type ns-name="CARD" id="116328" />
      <attribute name="Card ID" type="integer">
        <rec:attr ns-name="CARD_ID" attr-id="1234567" />
      </attribute>
      <attribute name="Parent ID" type="integer" relation="Card to Device">
        <rec:attr ns-name="PARENT_ID" attr-id="1234568" />
      </attribute>
      <attribute name="Name" type="string">
        <rec:attr ns-name="NAME" attr-id="1234569" />
      </attribute>
    </entity>
    <entity name="Device" pk="Device ID">
      <recobject-type ns-name="DEVICE" id="116329" />
      <attribute name="Device ID" type="integer" relation="Card to Device">
        <rec:attr ns-name="DEVICE_ID" attr-id="116329" />
      </attribute>
    </entity>
  </entity-set>
</configuration>

FIG. 61
### NC Object Types

<table>
<thead>
<tr>
<th>Object Type ID</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>CARD</td>
</tr>
<tr>
<td>2</td>
<td>DEVICE</td>
</tr>
</tbody>
</table>

### NC Attributes

<table>
<thead>
<tr>
<th>Attr ID</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>CARD_ID</td>
</tr>
<tr>
<td>2</td>
<td>PARENT_ID</td>
</tr>
<tr>
<td>3</td>
<td>NAME</td>
</tr>
</tbody>
</table>

### NC Objects

<table>
<thead>
<tr>
<th>Object ID</th>
<th>Name</th>
<th>Object Type ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>123456-888</td>
<td>116328</td>
</tr>
<tr>
<td>2</td>
<td>123000-753654</td>
<td>116328</td>
</tr>
<tr>
<td>3</td>
<td>Dev_1</td>
<td>116329</td>
</tr>
<tr>
<td>4</td>
<td>Dev_2</td>
<td>116329</td>
</tr>
</tbody>
</table>

### NC Params

<table>
<thead>
<tr>
<th>Attr ID</th>
<th>Object ID</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1234567</td>
<td>456</td>
</tr>
<tr>
<td>2</td>
<td>1234567</td>
<td>457</td>
</tr>
<tr>
<td>3</td>
<td>1234567</td>
<td>458</td>
</tr>
<tr>
<td>4</td>
<td>1234567</td>
<td>459</td>
</tr>
<tr>
<td>5</td>
<td>1234568</td>
<td>456</td>
</tr>
<tr>
<td>6</td>
<td>1234568</td>
<td>457</td>
</tr>
<tr>
<td>7</td>
<td>1234568</td>
<td>458</td>
</tr>
<tr>
<td>8</td>
<td>1234568</td>
<td>459</td>
</tr>
<tr>
<td>9</td>
<td>1234569</td>
<td>456</td>
</tr>
<tr>
<td>10</td>
<td>1234569</td>
<td>457</td>
</tr>
<tr>
<td>11</td>
<td>1234569</td>
<td>458</td>
</tr>
<tr>
<td>12</td>
<td>1234569</td>
<td>459</td>
</tr>
</tbody>
</table>

**FIG. 6J**
Unified Configuration

<configuration name="MyConfiguration"
 xmlns="http://www.netcracker.com/schemas/mediation/transition"
 xmlns:v="http://www.netcracker.com/schemas/mediation/transition/validation"
 xmlns:tr="http://www.netcracker.com/schemas/mediation/transition/
 transformation"
 xmlns:rec="http://www.netcracker.com/schemas/mediation/transition/
 reconciliation">
  <entity-set name="SDB">
    <entity name="BSCBoard">
      <attribute name="FDN" />
      <attribute name="Serial_Number" />
    </entity>
    <entity name="BSCFrame">
      <attribute name="FDN" />
      <attribute name="Serial_Number" />
    </entity>
  </entity-set>
  <entity-set name="IDB Entity Set">
    <rec:target-inventory name="Test Inventory Project" />
    <entity name="Card" pk="Card ID">
      <rec:object-type nc-name="Card" id="116328" />
      <attribute name="Card ID" type="integer">
        <tr:expression transformation="Device Transformation">
          <tr:source-attribute entity="BSCBoard" name="FDN" />
        </tr:expression>
        <rec:attr nc-name="CARD_ID" attr-id="1234567" />
      </attribute>
      <attribute name="Parent ID" type="integer" relation="Card to
 Device">
        <v:between majority="minor">
          <v:min>0</v:min>
          <v:max>1000000</v:max>
        </v:between>
        <v:is-not-null name="IsNotNull Validation Rule"
 majority="major" />
        <v:regexp-expression name="RegExp Validation Rule"
 majority="minor">
          <v:expression>\d</v:expression>
        </v:regexp-expression>
      </attribute>
      <attribute name="FDN" />
    </rec:object-type>
  </entity-set>
</configuration>

FIG. 7A
Unified Configuration 700b

```
<attribute name="Name" type="string">
    <tr:expression transformation="Device Transformation">
        merge-rule="FDN-2 || FDN-1">
            <tr:source-attribute entity="BSCFrame"
                name="FDN" alias="FDN-2" />
        
        <tr:source-attribute entity="BSCBoard" name="FDN"
            alias="FDN-1" />
    </tr:expression>

    <rec:attr nc-name="NAME" attr-id="1234569" />
    </attribute>

    <tr:source-attribute entity="BSFFrame" alias="FRAME" />
    <tr:source-attribute entity="BSCBoard" alias="BOARD" />
    
    <tr:join-condition name="By Serial Number">
        <tr:source-attribute entity="BOARD.Serial_Number" />
        <tr:source-attribute entity="FRAME.Serial_Number" />
    </tr:join-condition>

    <tr:filter name="Filter with like condition="like '%123%'">
        <tr:source-attribute entity="BOARD.FDN" />
    </tr:filter>

    <entity name="Device" pk="Device ID">
        <attribute name="Device ID" type="integer" relation="Card to Device" />
    </entity>

    <entity name="Device" pk="Device ID">
        <relationship name="Card to Device" multiplicity="0..1" direction="in">
            <source-entity>Card</source-entity>
            <destination-entity>Device</destination-entity>
        </relationship>
    </entity-set>
```

FIG. 7B
810a Transition Project

820a Unified Configuration

830a Entity Set

840a Entity

850a Attribute

835a EntitySet Rule

845a Entity Rule

855a Attribute Rule

FIG. 8A
FIG. 9
Generate Validation Rules 1130

Validate Input Data based on Validation Rules 1140

Generate Transformation Rules 1150

Transform Validated Data based on Transformation Rules 1160

Generate Reconciliation Rules 1170

Reconcile Transformed Data based on Reconciliation Rules 1180

Determine Validation Errors 1142

Modify the Input Data to Remove Validation Errors 1144

Generate Post-Transformation Validation Rules 1162

Validate Transformed Data based on Post-Transformation Validation Rules 1164

FIG. 11
Generate First Extraction Rules 1210a

Extract First Input Data based on First Extraction Rules 1220a

Generate First Transformation Rules 1230a

Transform First Input Data based on First Transformation Rules 1240a

Generate Second Extraction Rules 1210b

Extract Second Input Data based on Second Extraction Rules 1220b

Generate Second Transformation Rules 1230b

Transform Second Input Data based on Second Transformation Rules 1240b

Generate Combined Transformation Rules 1250

Generate Reconciliation Rules 1270

Transform Combined Input Data based on Combined Transformation Rules 1260

Reconcile Transformed Data based on Reconciliation Rules 1280

FIG. 12
DECLARATIVE AND UNIFIED DATA TRANSITION

FIELD OF THE INVENTION

[0001] The present invention relates generally to computer-based methods and apparatuses, including computer program products, for declarative and unified data transition.

BACKGROUND

[0002] Data transition, such as in system integration or ETL (extract, transform, and load) processes, is generally described as a list of steps of a whole process, i.e. in an imperative process. In implementing this list of steps, a knowledge domain is spread among all steps of the process. For example, in an extract step, where to put the data is described, in a validation step, what to check is described, and in a reconciliation step, what to transform is described. In other words, the same knowledge domain is partially described in every step of the data transformation. This repeating of knowledge definition(s) increases modeling time for the system, decreases reusability, and causes errors in knowledge consistency and integrity.

[0003] Thus, a need exists in the art for improved declarative and unified data transformation.

SUMMARY

[0004] One approach to declarative and unified data transition is a computer implemented method. The method includes determining an unified configuration for a knowledge domain, the unified configuration including one or more predicates for one or more system objects, one or more relationships between the one or more system objects, or any combination thereof; generating one or more transformation rules based on the one or more predicates; transforming input data based on the one or more transformation rules, the input data including information associated with the one or more system objects, the one or more relationships between the one or more system objects, or any combination thereof; determining one or more reconciliation rules based on the one or more predicates, the one or more reconciliation rules associated with a destination database; and reconciling the transformed data with the destination database based on the one or more reconciliation rules.

[0005] Another approach to declarative and unified data transition is a computer implemented method. The method includes determining an unified configuration for a knowledge domain, the unified configuration including one or more predicates for one or more system objects, one or more relationships between the one or more system objects, or any combination thereof; generating one or more transformation rules based on the one or more predicates, the one or more transformations enabling transformation of input data, the input data including information associated with the one or more system objects, the one or more relationships between the one or more system objects, or any combination thereof; and generating one or more reconciliation rules based on the one or more predicates, the one or more reconciliation rules associated with a destination database and enabling reconciliation of the transformed data with the destination database.

[0006] Another approach to declarative and unified data transition is a computer program product. The computer program product is tangibly embodied in an information carrier. The computer program product includes instructions being operable to cause a data processing apparatus to perform any one of the steps described herein.

[0007] Another approach to declarative and unified data transition is a system. The system includes an unified configuration module configured to determine an unified configuration for knowledge domain, the unified configuration including one or more predicates for one or more system objects, one or more relationships between the one or more system objects, or any combination thereof; a transformation rules engine configured to generate one or more transformation rules based on the one or more predicates; a transformation module configured to transform input data based on the one or more transformation rules, the input data including information associated with the one or more system objects, the one or more relationships between the one or more system objects, or any combination thereof; a reconciliation rules engine configured to generate one or more reconciliation rules based on the one or more predicates, the one or more reconciliation rules associated with a destination database; and a reconciliation module configured to reconcile the transformed data with the destination database based on the one or more reconciliation rules.

[0008] Another approach to declarative and unified data transition is a system. The system includes an unified configuration module configured to determine an unified configuration for a knowledge domain, the unified configuration including one or more predicates for one or more system objects, one or more relationships between the one or more system objects, or any combination thereof; a transformation rules engine configured to generate one or more transformation rules based on the one or more predicates, the one or more transformations enabling transformation of input data, the input data including information associated with the one or more system objects, the one or more relationships between the one or more system objects, or any combination thereof; and a reconciliation rules engine configured to generate one or more reconciliation rules based on the one or more predicates, the one or more reconciliation rules associated with a destination database and enabling reconciliation of the transformed data with the destination database.

[0009] Another approach to declarative and unified data transition is a system. The system includes means for determining an unified configuration for a knowledge domain, the unified configuration including one or more predicates for one or more system objects, one or more relationships between the one or more system objects, or any combination thereof; means for generating one or more transformation rules based on the one or more predicates; means for transforming input data based on the one or more transformation rules, the input data including information associated with the one or more system objects, the one or more relationships between the one or more system objects, or any combination thereof; and means for reconciling the transformed data with the destination database based on the one or more reconciliation rules.

[0010] Another approach to declarative and unified data transition is a system. The system includes means for determining an unified configuration for a knowledge domain, the unified configuration including one or more predicates for one or more system objects, one or more relationships between the one or more system objects, or any combination thereof; means for generating one or more reconciliation rules based on the one or more predicates, the one or more reconciliation rules associated with a destination database; and means for reconciling the transformed data with the destination database based on the one or more reconciliation rules.
thereof; means for generating one or more transformation rules based on the one or more predicates, the one or more transformations enabling transformation of input data, the input data including information associated with the one or more system objects, the one or more relationships between the one or more system objects, or any combination thereof; and means for generating one or more reconciliation rules based on the one or more predicates, the one or more reconciliation rules associated with a destination database and enabling reconciliation of the transformed data with the destination database.

In other examples, any of the approaches above can include one or more of the following features.

In some examples, the method further includes transforming second input data based on the one or more transformation rules, the second input data including information associated with the one or more system objects, the one or more relationships between the one or more system objects, or any combination thereof; and reconciling the second transformed data with the destination database based on the one or more reconciliation rules.

In other examples, the method further includes generating one or more validation rules based on the one or more predicates; and validating the input data based on the one or more validation rules.

In some examples, the method further includes automatically modifying the input data based on the validation of the input data to remove at least one validation error associated with the input data.

In other examples, the method further includes extracting the input data from one or more source databases based on one or more extraction rules.

In some examples, the method further includes generating the one or more extraction rules based on the one or more predicates.

In other examples, the extracting the input data from the one or more source databases and the reconciling the transformed data with the destination database occur relative to each other in real-time or substantially in real-time.

In some examples, the method further includes generating one or more second reconciliation rules based on the one or more predicates, the one or more second reconciliation rules associated with a second destination database; and reconciling the transformed data with the second destination database based on the one or more second reconciliation rules.

In other examples, the transforming the input data further including joining at least one part of the input data with at least one other part of the input data based on the one or more transformation rules.

In some examples, the transforming the input data further including merging at least one part of the input data with at least one other part of the input data based on the one or more transformation rules.

In other examples, the method further includes generating one or more post-transformation validation rules based on the one or more predicates; and validating the transformed data based on the one or more post-transformation validation rules.

In some examples, the method further includes merging the reconciled data with data stored on the destination database.

In other examples, the method further includes appending the reconciled data with data stored on the destination database.

In some examples, the input data including one or more table sets, one or more tables, one or more table fields, or any combination thereof.

In other examples, the destination database including an operational support system database, a healthcare system database, an information technology database, or any combination thereof.

In some examples, the method further includes generating one or more second transformation rules based on the one or more predicates; transforming second input data based on the one or more second transformation rules, the second input data including information associated with the one or more system objects, the one or more relationships between the one or more system objects, or any combination thereof; generating one or more combined transformation rules based on the one or more predicates; and transforming the first and second transformed input data based on the one or more combined transformation rules, the first and second transformed input data including information associated with the one or more system objects, the one or more relationships between the one or more system objects, or any combination thereof.

In other examples, the method further includes generating one or more combined reconciliation rules based on the one or more predicates; reconciling the transformed data with the destination database based on the one or more combined transformation rules.

In some examples, the system further includes the transformation module further configured to transform second input data based on the one or more transformation rules, the second input data including information associated with the one or more system objects, the one or more relationships between the one or more system objects, or any combination thereof; and the reconciliation module further configured to reconcile the second transformed data with the destination database based on the one or more reconciliation rules.

In other examples, the system further includes a validation rules engine configured to generate one or more validation rules based on the one or more predicates; and a validation module configured to validate the input data based on the one or more validation rules.

In some examples, the system further includes the validation module further configured to automatically modify the input data based on the validation of the input data to remove at least one validation error associated with the input data.

In other examples, the system further includes a source extraction module configured to extract the input data from one or more source databases based on one or more extraction rules.

In some examples, the system further includes a source extraction rule engine configured to generate the one or more extraction rules based on the one or more predicates.

In other examples, the system further includes the reconciliation rules engine further configured to generate one or more reconciliation rules based on the one or more predicates; and the reconciliation module further configured to reconcile the transformed
data with the second destination database based on the one or more second reconciliation rules.

[0034] In some examples, the system further includes the transformation module further configured to join at least one part of the input data with at least one other part of the input data based on the one or more transformation rules.

[0035] In other examples, the system further includes the transformation module further configured to merge at least one part of the input data with at least one other part of the input data based on the one or more transformation rules.

[0036] In some examples, the system further includes the transformation module further configured to modify at least one part of the input data based on the one or more transformation rules.

[0037] In other examples, the system further includes a validation rules engine configured to generate one or more post-transformation validation rules based on the one or more predicates; and a validation module configured to validate the transformed data based on the one or more post-transformation validation rules.

[0038] In some examples, the system further includes the reconciliation module further configured to merge the reconciled data with data stored on the destination database.

[0039] In other examples, the system further includes the reconciliation module further configured to append the reconciled data with data stored on the destination database.

[0040] In some examples, the system further includes the transformation rules engine further configured to generate one or more second transformation rules based on the one or more predicates; the transformation module further configured to transform second input data based on the one or more second transformation rules, the second input data including information associated with the one or more system objects, the one or more relationships between the one or more system objects, or any combination thereof; the transformation rules engine further configured to generate one or more combined transformation rules based on the one or more predicates; and the transformation module further configured to transform the first and second transformed input data based on the one or more combined transformation rules, the first and second transformed input data including information associated with the one or more system objects, the one or more relationships between the one or more system objects, or any combination thereof.

[0041] In other examples, the system further includes the reconciliation rules engine further configured to generate one or more combined reconciliation rules based on the one or more predicates, the one or more combined reconciliation rules associated with one or more destination databases; and the reconciliation module further configured to reconcile the first and second combined transformed input data with the one or more destination database based on the one or more combined reconciliation rules.

[0042] The declarative and unified data transformation techniques described herein can provide one or more of the following advantages. An advantage to the declarative and unified data transition is that the technology enables the configuration and documentation of various types of data transitions without additional coding, thereby increasing the efficiency of data transformation by enabling additional new knowledge domains without requiring changes to the already configured parts. Another advantage is that the technology is reusable across various data transitions with limited reconfiguration because of the unified configuration, thereby providing cost-effective integration solutions, such as data migration and system integration in various context, like inventory reconciliation, legacy system migration, or integration.

[0043] Another advantage is that after the unified configuration is configured for a data transition project, the system can generate project documentation (e.g., design specification, low-level requirements, etc.) based on the unified configuration, thereby saving costs compared to the traditional development method, where users draft all design specification, and then accordingly configure the steps of the data transition project.

[0044] Other aspects and advantages of the present invention will become apparent from the following detailed description, taken in conjunction with the accompanying drawings, illustrating the principles of the invention by way of example only.

BRIEF DESCRIPTION OF THE DRAWINGS

[0045] The foregoing and other objects, features, and advantages of the present invention, as well as the invention itself, will be more fully understood from the following description of various embodiments, when read together with the accompanying drawings.

[0046] FIG. 1 is a diagram of an exemplary system for data transition between a plurality of databases and/or systems;

[0047] FIG. 2 is a diagram of an exemplary mediation transition system;

[0048] FIG. 3 is a diagram of an exemplary telecom operational support system;

[0049] FIG. 4 is a flow diagram of an exemplary process for data transformation;

[0050] FIG. 5A is a flow diagram of an exemplary process for data extraction;

[0051] FIG. 5B is a flow diagram of another exemplary process for data transition;

[0052] FIG. 6A is exemplary input data;

[0053] FIG. 6B is an exemplary data definition of a unified configuration;

[0054] FIG. 6C is exemplary validation rules;

[0055] FIG. 6D is exemplary validated data;

[0056] FIG. 6E is exemplary transformation rules;

[0057] FIG. 6F is exemplary transformed data;

[0058] FIG. 6G is exemplary post-transformation validation rules;

[0059] FIG. 6H is exemplary validated transformed data;

[0060] FIG. 6I is exemplary reconciliation rules;

[0061] FIG. 6J is exemplary reconciled data;

[0062] FIGS. 7A and 7B illustrate an exemplary unified configuration;

[0063] FIG. 8A illustrates an exemplary structure of rules in a unified configuration for a data transition project;

[0064] FIG. 8B illustrates an exemplary structure of hierarchy rules in a unified configuration for a data transition project;

[0065] FIG. 9 is a flowchart of another exemplary process for data transition;

[0066] FIG. 10 is a flowchart of another exemplary process for data transition;

[0067] FIG. 11 is a flowchart of another exemplary process for data transition; and
FIG. 12 is a flowchart of another exemplary process for data transition.

DETAILED DESCRIPTION

Declarative and unified data transition technology enables a description of a knowledge domain to be made once and utilized a plurality of times for the transition of data. The technology can manage the knowledge domain for the transition process by enabling the knowledge domain to be defined by users and/or administrators of the technology (e.g., add information, edit information, delete information, setup a template, etc.) and/or by generating the knowledge domain based on other processes and/or relationships. The knowledge domain can be declaratively described in an unified configuration. The unified configuration can include one or more predicates (e.g., <relationship name="Frame to Device" multiplicity="0 . . 1" direction="in">, <relationship name="Card to Device" multiplicity="0 . . 1" direction="in">, etc.) for one or more system objects (e.g., components of a system, devices of a system, cards of a system, entities of a system, parts of a system, etc.) and/or one or more relationships between the one or more system objects (e.g., card A is related to card B, cards A and B are part of device I, etc.). The unified configuration of the knowledge domain can be, for example, organized in a multi-level hierarchy (e.g., a set of objects, a single object, a set of object attributes, single attributes, etc.).

The technology can centrally manage the knowledge domain for all of the components in the transition process. The components can include a validation step (e.g., check data against a set of validation rules, verify that data matches a particular criteria, etc.), a transformation step (e.g., convert information from a first database structure to a second database structure, convert data into an intermediate database format, etc.), and a reconciliation step (e.g., merge data to an existing destination database, merge data to a plurality of existing destination databases, overwrite data to an existing destination database, ignore data during the reconciliation to an existing destination database, append data to an existing destination database, etc.). In some examples, the components include a data extraction process and/or one or more validation steps (e.g., post-transformation validation step, post-reconciliation validation step, etc.). The transition process can occur in real-time or near real-time to enable the efficient transition of the data (i.e., the validation step, the transformation step, and the reconciliation step occur in real-time or near real-time).

The technology can include the unified configuration for the knowledge domain that includes the one or more predicates and/or the centralized knowledge base. The one or more predicates can be described utilizing metadata to describe a business domain (e.g., operational support system, information technology system, healthcare system, customer service system, enterprise resource planning system, accounting system, etc.) and/or include predicates about all involved objects (e.g., entities, components, parts, etc.) and their relationships. The centralized knowledge base can be utilized by the steps of the integration process, including the validation step, the transformation step, and/or the reconciliation step. Another advantage to the data transformation process is that the one or more predicates and/or the centralized knowledge base enable the technology to be configurable, scalable, and applicable to broad business cases (e.g., integration of two or more systems, mirroring of data between two or more systems, etc.).

FIG. 1 is a diagram of an exemplary system. The system includes a mediation transition system, an operational support system database, an operational support system, and a plurality of external system databases A, B, C, through Z (generally referred to as A). The mediation transition system, the operational support system database, the operational support system, and/or the plurality of external system databases can communicate via one or more connections (e.g., a direct network connection, an indirect network connection, an external storage connection, etc.). The mediation transition system can extract data from one or more of the plurality of external system databases, validate the extracted data, transform the validated data, and/or reconcile the validated transformed data with data stored on the operational support system database. The mediation transition system can extract, validate, transform, and/or reconcile the data based on a unified configuration of a knowledge domain (e.g., operational support system knowledge domain, information technology system knowledge domain, etc.).

Although FIG. 1 illustrates the operational support system database and the operational support system, the system can be utilized for any type of application system (e.g., information technology system, telephone system, network management system, healthcare system, etc.).

FIG. 2 is a diagram of an exemplary mediation transition system. The mediation transition system includes a source extraction rule engine, a source extraction module, a unified configuration module, a validation rules engine, a validation module, a transformation rules engine, a transformation module, a reconciliation rules engine, a reconciliation module, an input device, an output device, a display device, a communication module, a processor, and a storage device. The modules and devices described herein can, for example, utilize the processor to execute computer executable instructions and/or include a processor to execute computer executable instructions (e.g., an encryption processing unit, a field programmable gate array processing unit, etc.). It should be understood that the mediation transition system can include, for example, other modules, devices, and/or processors known in the art and/or varieties of the illustrated modules, devices, and/or processors.

The source extraction rule engine generates one or more extraction rules based on the one or more predicates. The source extraction rule engine can generate, determine, modify, create, define, etc. the one or more extraction rules based on the one or more predicates, the unified configuration, and/or other stored information (e.g., data from another unified configuration, data from another knowledge domain, etc.). The one or more extraction rules can include information that identifies tables, columns, rows, lines, identifiers, and/or roles where the input data is stored (e.g., user identification data is stored in the User_ID table, user names are stored in the User_Name column, etc.).

The source extraction module extracts input data (also referred to as source database (SDB)) from one or more source databases (e.g., plain text file, organized data, file of data, marked-up data, commercially available database system, enterprise database, relational database, non-relational database, XML-based database, read-only database, etc.).
non-standard meta-model driven database, etc.) based on one or more extraction rules (e.g., `SELECT * from User_ID; SELECT * from User_Name WHERE ID=0;` etc.).

[0077] The unified configuration module 213 determines an unified configuration for knowledge domain. The unified configuration includes one or more predicates for one or more system objects (e.g., doctor identification includes licensed states, patient identification includes mailing address, etc.) and/or one or more relationships between the one or more system objects (e.g., patient identification includes primary healthcare provider identification, doctor identification includes hospital identification, etc.).

[0078] The validation rules engine 214 generates one or more validation rules based on the one or more predicates and/or generates one or more post-transformation validation rules based on the one or more predicates. The validation rule engine 214 can generate (e.g., determine, modify, create, define, etc.) the one or more validation rules based on the one or more predicates, the unified configuration, and/or other stored information (e.g., data from another unified configuration, data from another knowledge domain, transformation rules, reconciliation rules, operational support system model, destination system model, database constraints, customized validations, etc.). For example, the validation rule engine 214 determines one or more validation rules based on the unified configuration associated with a destination database (e.g., specific syntax, specific data structure, etc.). The one or more validation rules can include number checks (e.g., integer check, minimum number, maximum number, etc.), null checks, key multiplicity, primary key validation, row-level validation, length validation, and/or any other type of validation (e.g., number, character, word, sentence, logical expression, etc.).

[0079] The validation module 215 validates the input data based on the one or more validation rules (e.g., is the data an integer, is the data above the number ten, etc.) and/or validates the transformed data based on the one or more post-transformation validation rules. The validation module 215 can automatically modify the input data based on the validation of the input data to remove at least one validation error associated with the input data (e.g., convert a written number to a numerically number, convert a real number to an integer, determine input for NULL data, delete the row/column/data, etc.). In other examples, the validation module 215 can report any validation errors.

[0080] The transformation rules engine 216 generates one or more transformation rules based on the one or more predicates. The transformation rule engine 216 can generate (e.g., determine, modify, create, define, etc.) the one or more transformation rules based on the one or more predicates, the unified configuration, and/or other stored information (e.g., data from another unified configuration, data from another knowledge domain, validation rules, reconciliation rules, operational support system model, destination system model, database constraints, customized transformations, etc.). The one or more transformations can enable the transformation of input data into transformed data (also referred to as intermediate database (IDB)). The input data can include information associated with the one or more system objects and/or the one or more relationships between the one or more system objects.

[0081] The transformation module 217 transforms input data based on the one or more transformation rules, transform second input data based on the one or more transformation rules, and/or transform any number of input data based on the one or more transformation rules. The second input data including information associated with the one or more system objects and/or the one or more relationships between the one or more system objects. The transformation module 217 can join at least one part of the input data with at least one other part of the input data based on the one or more transformation rules (e.g., join User table with Patient table, join User column with Doctor column, etc.). The transformation module 217 can merge at least one part of the input data with at least one other part of the input data based on the one or more transformation rules (e.g., merge User table with Patient table for all active Users, merge Doctor column with Hospital column for all Doctors grossing over $1 million per year, etc.). The transformation module 217 can modify at least one part of the input data based on the one or more transformation rules (e.g., delete a part of the input data, convert numbers, etc.).

[0082] The reconciliation rules engine 218 generates one or more reconciliation rules based on the one or more predicates. The one or more reconciliation rules can be associated with a destination database (e.g., plain text file, organized data, file of data, marked-up data, commercially available database system, enterprise database, relational database, non-relational database, XML-based database, write-once database, non-standard meta-model driven database, etc.). The reconciliation rules engine 218 can generate one or more second reconciliation rules based on the one or more predicates. The one or more second reconciliation rules can be associated with a second destination database. The one or more reconciliation rules and/or the one or more second reconciliation rules can enable reconciliation of the transformed data with the destination database.

[0083] The reconciliation module 219 reconciles the transformed data with the destination database based on the one or more reconciliation rules and/or reconciles the second transformed data with the destination database based on the one or more reconciliation rules. The reconciliation module 219 can reconcile the transformed data with the second destination database based on the one or more second reconciliation rules. The reconciliation module 219 can merge the reconciled data with data stored on the destination database (e.g., combine parts of the reconciled data with parts of the data stored on the destination database, transformation rules, validation rules, operational support system model, destination system model, database constraints, customized validations, etc.). The reconciliation module 219 can append the reconciled data with data stored on the destination database (e.g., add the parts of the reconciled data to the end of the data stored on the destination database, etc.).

[0084] The input device 221 receives information associated with the mediation transition system 210 (e.g., instructions from a user, instructions from another computing device, etc.) from a user (not shown) and/or another computing system (not shown). The input device 221 can include, for example, a keyboard, a scanner, etc. The output device 222 outputs information associated with the mediation transition system 210 (e.g., information to a printer (not shown), information to a speaker, etc.).

[0085] The display device 223 displays information associated with the mediation transition system 210 (e.g., status information, input data, transformed data, validated data, reconciled data, etc.). The communication module 224 receives the input data and/or transmits the reconciled data. The pro-
processor 225 executes the operating system and/or any other computer executable instructions for the mediation transition system 210 (e.g., executes applications, transforms data, reconciles data, security functions, etc.).

[0086] The storage device 226 stores input data, validated data, transformed data, reconciled data, instructions, and/or any other data associated with the mediation transition system 210. The storage device 226 can include a plurality of storage devices and/or the mediation transition system 210 can include a plurality of storage devices (e.g., an input data storage device, a transformed data storage device, etc.). The storage device 226 can include, for example, long-term storage (e.g., a hard drive, a tape storage device, flash memory, etc.), short-term storage (e.g., a random access memory, a graphics memory, etc.), and/or any other type of computer readable storage.

[0087] FIG. 3 is a diagram of an exemplar telecom operational support system 320. The telecom operational support system 320 can be, for example, the source or destination systems involved in data transition. The operational support system 320 includes a communication module 321, a database interface module 322, a service inventory module 323, a resource inventory module 324, an input device 331, an output device 332, a display device 333, a processor 334, and/or a storage device 335. The modules and devices described herein can, for example, utilize the processor 334 to execute computer executable instructions and/or include a processor to execute computer executable instructions (e.g., an encryption processing unit, a field programmable gate array processing unit, etc.). It should be understood that the operational support system 320 can include, for example, other modules, devices, and/or processors known in the art and/or varieties of the illustrated modules, devices, and/or processors.

[0088] The communication module 224 receives the reconciled data and/or transmits the output data (e.g., data utilized as input data by the mediation transition system, etc.). The database interface module 322 communicates with the storage device 335, an operational support system database, and/or one or more other external databases. The service inventory module 323 manages the service inventory for the operational support system 320 (e.g., receives updates to the service inventory, transmits updates to the service inventory, etc.). The resource inventory module 324 manages the resource inventory for the operational support system 320 (e.g., receives updates to the resource inventory, transmits updates to the resource inventory, etc.).

[0089] The input device 331 receives information associated with the operational support system 320 (e.g., instructions from a user, instructions from another computing device, etc.) from a user (not shown) and/or another computing system (not shown). The input device 331 can include, for example, a keyboard, a scanner, etc. The output device 332 outputs information associated with the operational support system 320 (e.g., information to a printer (not shown), information to a speaker, etc.).

[0090] The display device 333 displays information associated with the operational support system 320 (e.g., status information, output data, reconciled data, etc.). The processor 334 executes the operating system and/or any other computer executable instructions for the operational support system 320 (e.g., executes applications, transforms data, manages data, security functions, etc.).

[0091] The storage device 335 stores data, reconciled data, instructions, and/or any other data associated with the operational support system 320. The storage device 335 can include a plurality of storage devices and/or the operational support system 320 can include a plurality of storage devices (e.g., an input data storage device, a transformed data storage device, etc.). The storage device 335 can include, for example, long-term storage (e.g., a hard drive, a tape storage device, flash memory, etc.), short-term storage (e.g., a random access memory, a graphics memory, etc.), and/or any other type of computer readable storage.

[0092] FIG. 4 is a flow diagram 400 of an exemplary process for data transition. An external extraction module 420 extracts input data 428 from source data 410 (e.g., from external storage such as a text file, from a database, from an external device and system utilizing a protocol such as simple network management protocol (SNMP) and/or a command line interface (CLI), etc.). A unified configuration module (not shown) generates validation rules 434, transformation rules 444, and reconciliation rules 454 based on a data definition 474. The validation rules 434, the transformation rules 444, the reconciliation rules 454, and the data definition 474 are an unified configuration 470.

[0093] A validation module 430 validates the input data 428 to form validated data 438 based on the validation rules 434. A transformation module 440 transforms the validated data 438 to transformed data 448 based on the transformation rules 444. A reconciliation module 450 reconciles the transformed data 448 into a destination data 460 via reconciled data 458 based on the reconciliation rules 454.

[0094] In some examples, the unified configuration 470 describes system objects (e.g., entities, components, etc.) and their relationships. The unified configuration 470 can specify extra parameters for the system objects which add information in one or more knowledge domains. The knowledge domains can be stored in a configuration in an unified independent manner (e.g., for XML, each knowledge domain is in a different XML namespace). The knowledge domain can include one or more predicates that provide information regarding the behavior of different processes. For example, the predicates includes information about how to obtain data for an entity, information about how to check consistency, and information about how to load data into a destination database. Each process step can define how it interprets data in a specific knowledge domain. The unified configuration can include extra declarations and/or create special processes for the extra declarations. For example, the extra declarations include how to load data into database tables from simple network management protocol (SNMP) agents or lightweight directory access protocol (LDAP).

[0095] In other examples, the unified configuration 470 enables a user and/or an administrator to describe once the domain knowledge of the transition, in one place, and then map it to source database, to the destination database, and/or rules for checking consistency of the data. The technology described herein can be utilized for various types of integration and/or migration, such as mapping a data model to SNMP, mapping a read-only database to a legacy element management service (EMS), etc.

[0096] FIG. 5A is a flow diagram 500A of an exemplary process for data extraction. An external extraction module 520A extracts input data 528A from source data 510A based on extraction rules 535A. A source extraction rule engine (not shown) generates the extraction rules 535A based on a data...
The extraction rules 535a and the data definition 574a are generally described as an unified configuration 570a.

FIG. 53 is a flow diagram of another exemplary process for data extraction. A validation rule engine, transformation rule engine, and a reconciliation rule engine (not shown) generate validation rules 546b, transformation rules 544b, and reconciliation rules 554b, respectively, based on a data definition 574b. The validation rules 546b, the transformation rules 544b, the reconciliation rules 554b, and the data definition 574b are parts of the unified configuration 570b.

A transformation module 540b transforms input data 528b to transformed data 548b based on the transformation rules 544b. A validation module 545b validates the transformed data 548b to form valid transformed data 549b based on the validation rules 546b. A reconciliation module 550b reconciles the validated transformed data 549b into a destination data 560b via reconciled data 558b based on the reconciliation rules 554b.

FIG. 6A is exemplary input data 600a. The input data 600a includes SBCBoard data 610a, BCFFrame data 620a, and BSDevice data 630a. The input data 600a can be extracted from a source database (e.g., plain text file, a hierarchical database system, etc.) based on one or more extraction rules.

FIG. 6B is an exemplary unified configuration 600b. The unified configuration 600b includes a data definition part. The unified configuration 600b includes source database (SDB) configuration 610b and intermediate database (IDB) configuration 620b. The unified configuration 600b describes a knowledge domain (KD). The unified configuration 600b can include system objects (e.g., entities such as SBC Board and/or Card, etc.) associated with the KD, attributes of the system objects, relationships between system objects, and/or multiplicities of the relationships. The system object set (e.g., entity set) can be a container that provides capabilities to specify system object settings and references that utilized to prepare the validation, the transformation, and/or the reconciliation process. The system object can include one or more attributes (e.g., a primary key attribute which is a unique attribute which identifies the system object). Every attribute can be defined by the attribute type. Attributes from different system objects can be involved in relationships with uni-, in- and/or out-directions.

FIG. 6C is exemplary validation rules 600c. The validation rules 600c include the SDB configuration as illustrated above in the unified configuration 600b and three validation rules 610c, 620c, and 630c. The validation rules engine 214 of FIG. 2 can generate (e.g., determine, find, modify, etc.) the three validation rules 610c, 620c, and 630c based on the unified configuration 600b and/or other information associated with the knowledge domain (e.g., rules defined by an administrator, rules utilized in a previous validation of the same knowledge domain, rules defined in the knowledge domain, etc.).

In some examples, the validation rules can be defined on every level of the KD (e.g., attribute, entity, entity-set, etc.). The attribute-level validation rules 610c, 620c, and 630c check values of concrete attribute on a given entity. In some example, the “Between” rule checks whether the attribute value is in the range specified by the values of the “min” and “max” parameters. In other examples, the “Is Not Null” rule checks whether the attribute value is not null. In some examples, the “Regular Expression” rule checks the attribute value against the mask specified by the value of the “expression” parameter.

FIG. 6D is exemplary validated data 600d. The validation data 600d is the result of validation based on the validation rules 600c. The validated data 600d includes four deleted entries 610d, 620d, 630d, and 640d. The first deleted entry 610d is deleted based on the “IsNotNull 1 Validation Rule” 610c, the second deleted entry 620d is deleted based on the “IsNotNull 2 Validation Rule” 620c, the third deleted entry 630d and fourth deleted entry 640d are deleted based on the “Like Validation Rule” 630c.

FIG. 6E is exemplary transformation rules 600e. The transformation rules 600e include a plurality of transformation rules 610e, 620e, 630e, 640e, 650e, and 660e. The transformation engine 216 of FIG. 2 can generate the transformation rules 610e, 620e, 630e, 640e, 650e, and 660e based on the unified configuration 600b and/or other information associated with the knowledge domain (e.g., rules defined by an administrator, rules utilized in a previous transformation of the same knowledge domain, etc.). As illustrated, the transformation rule 630e merges BSCFrame Serial Number and the BSCCard Serial Number together to form the Name field for the Card entity.

In some examples, the transformation rules are defined on every level of the KD. The transformation rules engine 216 can define one “transformation” session from SDB to IDB and one transformation expression (rule) on the attribute “Name.” Every transformation can be defined for source entities in the SDB. In other examples, destination expressions define mapping among source and destination attributes and/or can express destination attribute value using special merge-rule. In some examples, a join condition is the equal condition of two source entities for joining into result. In other examples, data from source entities can be filtered using special filter by attribute value.

FIG. 6F is exemplary transformed data 600f. The validated data 600d is transformed into the transformed data 600f based on the transformation rules 600e. The transformed data 600f includes card data 610f and device data 620f.

FIG. 6G is exemplary post-transformation validation rules 600g. The post-transformation validation rules 600g include two validation rules 610g and 620g. The validation rules engine 214 of FIG. 2 can generate (e.g., determine, find, modify, etc.) the validation rules 610g and 620g based on the unified configuration 600b and/or other information associated with the knowledge domain (e.g., rules defined by an administrator, rules utilized in a previous validation of the same knowledge domain, etc.).

FIG. 6H is exemplary validated transformed data 600h. The transformed data 600f is validated into the validated transformed data 600h based on the post-transformation validation rules 600g. The validation module 215 of FIG. 2 deletes one entry 610h. The deleted entry 610h is deleted based on the “RegExp Validation Rule” 620g.

FIG. 6I is exemplary reconciliation rules 600i. The reconciliation rules 600i include a plurality of reconciliation rules 610i, 620i, 630i, 640i, 650i, and 660i. The reconciliation rules engine 218 of FIG. 2 can generate (e.g., determine, find, modify, etc.) the reconciliation rules 610i, 620i, 630i, 640i, 650i, and 660i based on the unified configuration 600b and/or other information associated with the knowledge domain (e.g., rules defined by an administrator, rules utilized in a previous reconciliation of the same knowledge domain, etc.).
In some examples, the reconciliation rules $600i$ can be defined on every level of the KD. The reconciliation rules can include three reconciliation mapping rules. In other examples, every entity set is reconciled to an inventory project, every entity is mapped to a special object type, and/or every attribute is mapped to a special attribute.

Although FIGS. 6A through 6J illustrate the data definition $600b$ of the unified configuration and rules utilizing extensible markup language (XML), the data definition $600b$ and/or one or more of the rules can utilize any type of protocol, language, and/or standard (e.g., plain text), spreadsheet, relational or other database, Java Script, Lisp, Haskell, etc.

Although FIGS. 7A and 7B illustrate an exemplary unified configuration $700a$ and $700b$. The unified configuration $700a$ and $700b$ includes the IDB section $710a$ and the SDB section $720a$ and $720b$. As illustrated in FIGS. 7A and 7B, the unified configuration can be a single configuration file that is utilized by the modules and/or engines described herein.

Although FIGS. 7A and 7B illustrate the unified configuration $700a$ and $700b$ with a single IDB section $710a$ and a single SDB section $720a$. The unified configuration $700a$ and $700b$ can include a plurality of IDB sections and/or a plurality of SDB sections. The technology can utilize part or all of the plurality of IDB sections and/or the plurality of SDB sections.

In some examples, the IDB section $710a$ represents parameters of a particular entity (e.g., a circuit, a device, etc.). If the particular entity includes multiple values, the multiple values can be represented in a parameter table.

FIG. 8A illustrates an exemplary structure $800a$ of rules for data transition. The structure $800a$ includes a transition project $810a$, a unified configuration $820a$, an entity set $830a$, an entity $840a$, and an attribute $850a$. The entity set level $830a$ includes entity sets $835a$. The entity level $840a$ includes entity rules $845a$. The attribute level $850a$ includes attribute rules $855a$. The rules and/or structure thereto can be utilized by the modules and/or engines described herein to generate extraction rules, validation rules, transformation rules, and/or reconciliation rules.

FIG. 8B illustrates an exemplary structure $800b$ of hierarchy rules for data transition. The structure $800b$ includes a transition rule $810b$. The transition rule $810b$ includes an entity set rule $820b$, an entity rule $830b$, and an attribute rule $840b$. The entity set rule $820b$ includes an integration entity set rule $825b$. The entity rule $830b$ includes an integration entity rule $835b$. The attribute rule $840b$ includes an integration entity attribute rule $845b$. The rules and/or structure thereto can be utilized by the modules and/or engines described herein to generate extraction rules, validation rules, transformation rules, and/or reconciliation rules.

FIG. 9 is a flowchart 900 of another exemplary process for data transition utilizing, for example, the media
tion transition system 210 of FIG. 2. The source extraction rule engine 211 generates (910a, 910b through 910e) the one or more first, second through Nth extraction rules based on the one or more predicates. The source extraction module 212 extracts (920a, 920b through 920c) the first, second through Nth input data from one or more source databases based on the first, second through Nth extraction rules, respectively.

The validation rules engine 214 generates (930) one or more validation rules based on the one or more predicates. The validation module 215 validates (940) the input data (in this example, the first, second through Nth input data) based on the one or more validation rules. The transformation engine 216 generates (950) one or more transformation rules based on the one or more predicates. The transformation module 217 transforms (960) input data based on the one or more transformation rules. The reconciliation rules engine 218 generates (970) one or more reconciliation rules based on the one or more predicates, and the one or more reconciliation rules associated with a destination database. The reconciliation module 219 reconciles (980) the transformed data with the destination database based on the one or more reconciliation rules.

FIG. 10 is a flowchart 1000 of another exemplary process for data transition utilizing, for example, the mediation transition system 210 of FIG. 2. The validation rules engine 214 generates (1030) one or more validation rules based on the one or more predicates. The validation module 215 validates (1040) the input data based on the one or more validation rules. The transformation rules engine 216 generates (1050) one or more transformation rules based on the one or more predicates. The transformation module 217 transforms (1060) input data based on the one or more transformation rules. The reconciliation rules engine 218 generates (1070a, 1070b through 1070c) one or more first, second through Nth reconciliation rules based on the one or more predicates. Each of the one or more reconciliation rules are associated with a destination database (e.g., the one or more first reconciliation rules are associated with a first destination database, the one or more second reconciliation rules are associated with a second destination database, etc.). The reconciliation module 219 reconciles (1080a, 1080b through 1080c) the transformed data with the first, second through Nth destination databases based on the one or more first, second through Nth reconciliation rules, respectively.

Although FIG. 10 illustrates a flowchart 1000 of the reconciliation rules generation and execution as concurrent, the reconciliation rules generation and/or execution can occur in any order. For example, the reconciliation rules generation and/or execution are defined based on the destination databases. In other examples, the reconciliation rules generation can occur in a specified order, a runtime order, and/or based on prior reconciliation rules generations. In this example, the reconciliation execution can occur based on the reconciliation rule generation.

FIG. 11 is a flowchart 1100 of another exemplary process for data transition utilizing, for example, the mediation transition system 210 of FIG. 2. The validation rules engine 214 generates (1130) one or more validation rules based on the one or more predicates. The validation module 215 validates (1140) the input data based on the one or more validation rules. The validation module 215 determines (1142) one or more validation errors associated with the input data based on the one or more validation rules and/or modifies (1144) the input data to remove the one or more validation errors.
The transformation rules engine 216 generates (1050) one or more transformation rules based on the one or more predicates. The transformation module 217 transforms (1060) input data based on the one or more transformation rules. The validation rules engine 214 generates (1162) one or more post-transformation validation rules. The validation module 215 validates (1264) the transformed data based on the one or more post-transformation validation rules. The reconciliation rules engine 218 generates (1070) one or more reconciliation rules based on the one or more predicates. The reconciliation module 219 reconciles (1080) the transformed data with the destination databases based on the one or more reconciliation rules.

FIG. 12 is a flowchart 1200 of another exemplary process for data transition utilizing, for example, the mediation transition system 210 of FIG. 2. The source extraction rule engine 211 generates (1210a and 1210b) one or more first and second extraction rules based on the one or more predicates (e.g., concurrently, serially, dependent on each other, etc.). The source extraction module 212 extracts (1220a and 1220b) the first and second input data from one or more source databases based on the first and second extraction rules, respectively (e.g., concurrently, serially, dependent on each other, etc.).

The transformation rules engine 216 generates (1230a and 1230b) one or more first and second transformation rules based on the one or more predicates. The transformation module 217 transforms (1240a and 1240b) the first and second input data based on the one or more first and second transformation rules, respectively. The transformation rules engine 216 generates (1250) one or more combined transformation rules based on the one or more predicates. The one or more combined transformation rules are associated with the combined first and second input data (e.g., logically combined, linked together, associated together, etc.). The transformation module 217 transforms (1260) the combined first and second input data based on the one or more combined transformation rules.

The reconciliation rules engine 218 generates (1270) one or more reconciliation rules based on the one or more predicates. The reconciliation module 219 reconciles (1280) the transformed combined data with the destination databases based on the one or more reconciliation rules.

For example, the transition process described in FIG. 12 transitions parts of the first data and the second data with an OSS database and parts of the first data and the second data with a customer billing database from a customer service database. In other words, in this example, source data from a plurality of sources is transitioned to a destination database. In other examples, source data from a plurality of sources is transitioned to a plurality of destination databases.

Although FIG. 12 does not illustrate validation, the exemplary process can include validation as described herein. In some examples, the validation rules engine 214 generates one or more first validation rules based on the one or more predicates. The validation module 215 validates the first input data based on the one or more first validation rules. The validation module 215 determines one or more first validation errors associated with the first input data based on the one or more first validation rules and/or modifies the first input data to remove the one or more second validation errors.

In other examples, the validation rules engine 214 generates one or more second validation rules based on the one or more predicates. The validation module 215 validates the second input data based on the one or more second validation rules. The validation module 215 determines one or more second validation errors associated with the second input data based on the one or more second validation rules and/or modifies the second input data to remove the one or more second validation errors.

In some examples, the validation rules engine 214 generates one or more first and second post-transformation validation rules. The validation module 215 validates the first and second transformed data based on the one or more first and second post-transformation validation rules, respectively.

In other examples, the transformation rules generation and execution, the validation rules generation and execution, the reconciliation rules generation and execution, and/or other data transition operations can occur in any order and/or any sequence utilizing the unified configuration. For example, the data transition can occur as follows: input data from four source databases is extracted based on two sets of one or more extraction rules; the input data from each source database is validated based on a single set of one or more validation rules; the validated data from each source database is transformed based on two sets of one or more transformation rules; the two sets of transformed data is further transformed based on one set of one or more combined transformation rules; the combined set of transformed data is reconciled with two destination databases based on a single set of one or more reconciliation rules; and the transformed data from each source database is reconciled to four different destination databases based on four different sets of one or more reconciliation rules. In this example, the data is transitioned between a plurality of source databases to a plurality of destination databases based on various rules and/or in various orders. Other examples of the number of rules and/or sequence of the order of the rule generation and/or execution can be utilized by the technology described herein.

In other examples, the technology enables the reuse of the knowledge domain via use of declarative style in configuration definition, thereby reducing the need for custom programming and increasing the efficiency of the transformation process. In other words, in some examples, the knowledge domains and rules are not coded inside a module and/or engine. The knowledge domains and/or rules can be used by other module and/or engine. In other examples, the modules and/or engines can use part of the unified configuration based on the need of the module/engine (e.g., use only SONET/SDH domain and ignore VPN domain, etc.).

In some examples, the knowledge domain can be described regardless of its specific implementations (e.g., physical implementation, logical implementation, etc.). In this example, rules can be reused for different transformation projects, regardless if the rules are from Excel to MS SQL, Oracle to DB2, or MySQL to Informix, etc.

In other examples, the technology includes knowledge domain and/or rules management tools. The tools can enable easy configuration and/or export/import in any representation format. The technology can include a configuration editor. The configuration editor can include a knowledge domain editor, a rules editor, and/or a "Meta-rules" editor for validating of rules.

In some examples, the validation rules can include one or more entities definition checking (e.g., primary key existence, no duplicated attributes, etc.), relationships integrity checking (e.g., no duplication of relations, no circle relations with multiplicity impact, etc.), validation rules checking
against schema definition (e.g., no contradiction of rules in schema and in entity set, etc.), and/or reconciliation rules checking against integrity with schema (e.g., existence of all required attributes and object-types, no cycle in parent-child relationships, etc.).

[0137] In other examples, the technology described herein can synchronize data between two or more data sources (e.g., databases, files, etc.). For example, the technology can be utilized to synchronize customer data between a customer relationship system and an operational support system. In some examples, the one or more destination databases can be, for example, any type (e.g., a relational DB, an object DB, an XML-based DB, a metadata-driven database, etc.).

[0138] In some examples, the technology described herein can utilize the following exemplary data definition: input data (SDB), validation rules, validated input data, transformation rules, transformed data, post-transformation validation rules, validated transformed data, reconciliation rules, and reconciled data.

[0139] Exemplary Unified Configuration (in XML Notation)

```
<!-- Specifies 3 SDB tables (entities): BSCBoard, BSCBTSBoard and BSCDevice. -->
<configuration name="MyConfiguration"
 xmlns="http://www.netcracker.com/schemas/mediation/transformation"
 xmlns:r="http://www.netcracker.com/schemas/mediation/transformation/validation"
 xmlns:tr="http://www.netcracker.com/schemas/mediation/transformation/reconciliation">
<entity-set name="SDB">
<entity name="BSCBoard">
<attribute name="FDN"/>
<attribute name="Serial_Number"/>
<attribute name="Device" relation="Card to Device">
<vis-not-null name="IsNotNull 1 Validation Rule" majority="true"/>
</attribute>
</entity>
<entity name="BSCBTSBoard">
<attribute name="FDN"/>
<attribute name="Serial_Number"/>
<attribute name="Device" relation="Card to Device">
<vis-not-null name="IsNotNull 2 Validation Rule" majority="true"/>
</attribute>
</entity>
<entity name="BSCDevice">
<attribute name="Device"/>
<attribute name="Name"/>
</entity>
</entity-set>
<rectangle inventory name="Test Inventory Project">
<entity name="Card" pk="Card ID">
<recoobject-type nc-name="CARD" id="39507">
<attribute name="Card ID">
<tranexpression transformation="Card Transformation">
<attrsource attribute entity="BSCBoard" name="FDN"/>
</tranexpression>
<vmin>0<vmax>1000</vmax>
<vmax>1000</vmax>
</attribute>
<attribute name="Parent ID" relation="Card to Device">
<tranexpression transformation="Card Transformation">
<attrsource attribute entity="BSCBoard" name="Device"/>
</tranexpression>
<attribute name="Name" type="string">
<tranexpression transformation="Card Transformation" merge-rules="Card...">
Exemplary Input Data

BSCBoard

<table>
<thead>
<tr>
<th>FDN</th>
<th>Serial Number</th>
<th>Device</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>77221</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>77222</td>
<td>NULL</td>
</tr>
<tr>
<td>3</td>
<td>77223</td>
<td>3</td>
</tr>
<tr>
<td>4</td>
<td>77224</td>
<td>2</td>
</tr>
</tbody>
</table>

Comment: BSCBoard is a kind of motherboard. Every entity of this type can include a unique identifier on BSCBoard level (in this example, FDN), a unique identifier on entity set level (in this example, Serial Number), and/or a link to a Device.

BSCBTSBoard

<table>
<thead>
<tr>
<th>FDN</th>
<th>Serial Number</th>
<th>Device</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>77225</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>77226</td>
<td>3</td>
</tr>
<tr>
<td>3</td>
<td>77227</td>
<td>NULL</td>
</tr>
<tr>
<td>4</td>
<td>77228</td>
<td>1</td>
</tr>
</tbody>
</table>

Comment: BSCBTSBoard is another kind of motherboard. Every entity of this type can include a unique identifier on BSCBTSBoard level (in this example, FDN), a unique identifier on entity set level (in this example, Serial Number) and a link to Device.
Comment: The BSCDevice table includes information about devices: unique identifier (in this example, Device id) and a name of the device (in this example, Name).

Exemplary Validation Rules

<!——The validation rules define the validation of the source data. The validation rules specify 2 field level validation rules of type “Is Not Null”. These rules check values of attributes Device in BSCBoard and Device in BSCBTSBoard and delete all rows where the value is NULL.—>
Exemplary Validated Input Data (Validated SDB)

<table>
<thead>
<tr>
<th>FDN</th>
<th>Serial Number</th>
<th>Device</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>77221</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>NULL</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>77223</td>
</tr>
<tr>
<td>4</td>
<td>4</td>
<td>77224</td>
</tr>
</tbody>
</table>

Deleted according to "IsNotNull 1 Validation Rule"
Exemplary Transformation Rules

The transformation rules can, for example, specify 2 IDB tables (entities) Card and Device. Every row in table Card can include information about identifier (Card ID), parent device (Parent ID), and/or card name (Name). Every row in table Device can include information about name and identifier of device (Name and Device ID columns).

<table>
<thead>
<tr>
<th>Device</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Dev_1</td>
</tr>
<tr>
<td>2</td>
<td>Dev_2</td>
</tr>
<tr>
<td>3</td>
<td>Dev_3</td>
</tr>
<tr>
<td>4</td>
<td>Device-11</td>
</tr>
</tbody>
</table>

```xml
<configuration name="MyConfiguration" xmlns="http://www.netcracker.com/schemas/mediation/transition"
xmlns:xsi="http://www.netcracker.com/schemas/mediation/transition/validation"
  <entity-set name="IDB Entity Set">
    <entity-name>"BSCBoard"</entity-name>
    <attribute name="FDN"/>
    <attribute name="Serial_Number"/>
    <attribute name="Device" relation="Card to Device">
      <via name="IniNull 1 Validation Rule" major="major"/>
    </attribute>
  </entity>
  <entity name="BSCBTSBoard">
    <attribute name="FDN"/>
    <attribute name="Serial_Number"/>
    <attribute name="Device" relation="Card to Device">
      <via name="IniNull 2 Validation Rule" major="major"/>
    </attribute>
  </entity>
  <entity name="BSCDevice">
    <attribute name="Device"/>
    <attribute name="Name"/>
  </entity>
  <relationship name="Board to Device" multiplicity="0..1" direction="in">
    <source-entity>BSCBoard</source-entity>
    <destination-entity>Device</destination-entity>
  </relationship>
  <relationship name="Frame to Device" multiplicity="0..1" direction="in">
    <source-entity>BSCBTSBoard</source-entity>
    <destination-entity>Device</destination-entity>
  </relationship>
</entity-set>
<entity-set name="IDB Entity Set">
  <rec-target-inventory-name>"Test Inventory Project"</rec-target-inventory-name>
  <entity name="Card" pk="Card ID">
    <rec-object-type>"CARD"</rec-object-type>
    <attr name="CARD ID" attr-id="39500"/>
    <attribute name="Card ID">
      <traexpression transformations="Card Transformation">
        <tsource-attribute entity="BSCBoard" name="FDN"/>
      </traexpression>
      <vbetween majority="minor">
        <vmin>
        </vmin>
        <vmax>10000</vmax>
      </vbetween>
      <vregex expression="RegExp Validation Rule" major="major">
        <regex expression="[digit]\"/>
      </vregex>
      <vregex expression="[digit]\"/>
      <recattr nc-names="CARD_ID" attr-id="39500"/>
    </attribute>
    <attribute name="Parent ID" relation="Card to Device">
```
 Transformation SDB to IDB: This section defines process of transformation source tables to destination intermediate tables. It specifies 2 transformations (one on each destination table) “Card Transformation” and “Device Transformation”. Each column of IDB tables has special transformation level attribute (tr:expression) which defines source columns. 

<entity-set name="SDB">
<entity name="BSCBoard">
  <attribute name="FDN">
  </attribute>
  <attribute name="Serial_Number">
  </attribute>
  <attribute name="Device" relation="Card to Device">
    <not-null name="ISNotNull 1 Validation Rule" majority="major"/>
  </attribute>
</entity>
<entity name="BSCBTSBoard">
  <attribute name="FDN">
  </attribute>
  <attribute name="Serial_Number">
  </attribute>
  <attribute name="Device" relation="Card to Device">
    <not-null name="ISNotNull 2 Validation Rule" majority="major"/>
  </attribute>
</entity>
<entity name="BSCDevice">
  <attribute name="Device">
  </attribute>
  <attribute name="Name">
  </attribute>
</entity>
<relationship name="Board to Device" multiplicity="0..1" direction="in">
  <source-entity name="BSCBoard"/>
  <destination-entity name="Device"/>
</relationship>
<relationship name="Frame to Device" multiplicity="0..1" direction="in">
  <source-entity name="BSCBTSBoard"/>
  <destination-entity name="Device"/>
</relationship>
</entity-set>
<entity-set name="IDB Entity Set">
  <rectarget-inventory name="Test Inventory Project"/>
  <entity name="Card" pk="Card ID">
    <recobject type="nc-name" name="CARDID" id="39507">
      <attribute name="Card ID">
        <trexpression name="Card Transformation" transform="Card Transformation">
          <tresource-attribute entity="BSCBoard" name="FDN"/>
          <between minor="0" major="1000"/>
        </trexpression>
        <attr"
      </attribute>
      <attribute name="Parent ID" relation="Card to Device">
        <trexpression name="Card Transformation" transform="Card Transformation">
          <tresource-attribute entity="BSCBoard" name="Device"/>
        </trexpression>
        <attr">
      </attribute>
      <attribute name="Name" type="string">
        <trexpression name="Card Transformation" transform="Card Transformation">
          <tresource-attribute entity="BSCBoard" name="Serial Number">
            <attr">alias="SN"</attr>
          </tresource-attribute>
          <tresource-attribute entity="BSCBTSBoard" name="Serial Number">
            <attr">alias="F_SN"</attr>
          </tresource-attribute>
          <tresource-attribute entity="BSCBoard" name="Serial Number">
            <attr">alias="B_SN"</attr>
          </tresource-attribute>
        </trexpression>
        <attr">
      </attribute>
    </recobject>
  </entity>
</entity-set>
Exemplary Transformed Data (IDB)

Card—The Card table created via transformation of source tables BSCBoard and BSCBTSBoard.

<table>
<thead>
<tr>
<th>Card ID</th>
<th>Parent ID</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>Card_77221-77228 Card ID = BSCBoard.FDN Parent ID = BSCBoard.Device Name = Card + BSCBoard.Serial Number + - + BSCBTSBoard.Serial Number (where BSCBoard.Device = BSCBTSBoard.Device)</td>
</tr>
<tr>
<td>2</td>
<td>3</td>
<td>Card_77223-77226</td>
</tr>
<tr>
<td>3</td>
<td>4</td>
<td>Card_77224-77225</td>
</tr>
</tbody>
</table>

Device—The device table created via transformation of source table BSCDevice.

<table>
<thead>
<tr>
<th>Device ID</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Dev_1</td>
</tr>
<tr>
<td>2</td>
<td>Dev_2</td>
</tr>
<tr>
<td>3</td>
<td>Dev_3</td>
</tr>
<tr>
<td>4</td>
<td>Device-11</td>
</tr>
</tbody>
</table>

Exemplary Post-Transformation Validation Rules

<!—The post-transformation validation rules define validation of transformed data (IDB). This process of validation is similar to Validation of SDB. The post-transformation validation rules specify 3 validation rules “Between Validation Rule”, “RegExp Validation Rule” and “Like Validation Rule”. Each validation rule checks values in corresponded columns in accordance with defined expression. For example, the RegExp Validation Rule” requires that value of column “Card ID” should be only number type, because expression equals ![[:digit:]]—> <configuration name=“MyConfiguration”>
[0152] Exemplary Validated Transformed Data

**Card**

<table>
<thead>
<tr>
<th>Card ID</th>
<th>Parent ID</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>Card_77221-77228</td>
</tr>
<tr>
<td>2</td>
<td>3</td>
<td>Card_77223-77226</td>
</tr>
<tr>
<td>3</td>
<td>4</td>
<td>Card_77224-77225</td>
</tr>
</tbody>
</table>

All values are satisfied to validation rules “Between validation rule” and “RegExp Validation Rule”.

[0154] Device

<table>
<thead>
<tr>
<th>Device ID</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Dev_1</td>
</tr>
<tr>
<td>2</td>
<td>Dev_2</td>
</tr>
<tr>
<td>3</td>
<td>Dev_3</td>
</tr>
<tr>
<td>4</td>
<td>Dev_4</td>
</tr>
</tbody>
</table>

Deleted according to the “Like Validation Rule”.

[0155] Exemplary Reconciliation Rules

The reconciliation rules specify, on top level, 2 IDB tables (entities) Card and Device. Each entity is reconciled, correspondingly, rows of table “NC_OBJECTS” in destination DB, by specifying xml attribute `ac-name` "CARD" and DEVICE. Then, under each mapping between IDB entity and object-type, the reconciliation rules specify which attribute from IDB reconciled with what attribute in destination DB. Some columns in `NC_OBJECTS` for certain rows can be filled automatically without binding to the destination attribute. —>

```xml
<configuration name="MyConfiguration">
  <vis-null name="IsNotNull"/>
  <attributes
    <entity-set name="CARD">
      <entity name="BSCBoard">
        <attribute name="FDN"/>
        <attribute name="Serial_Number"/>
        <attribute name="Device" relatin="Card to Device">
          <vis-not-null name="IsNotNull 1 Validation Rule" majority="major"/>
        </attribute>
      </entity>
    </entity-set>
  </attributes>
</configuration>
```
<attribute name="FQN"/>
<attribute name="Serial_Number"/>
<attribute name="Device" relation="Card to Device">
  <visa-not-null name="IsNotNull 2 Validation Rule" majority="major"/>
</attribute>
</entity>
<entity name="BSCTarget">
  <attribute name="Device"/>
  <attribute name="Name"/>
  <relationship name="BSCTarget to Device" multiplicity="0..1" direction="in">
    <source-entity name="BSCTarget" />
    <destination-entity name="Device" />
  </relationship>
</entity>
<entity-set name="BSCTarget Entity Set">
  <rectarget-inventory name="Test Inventory Project" />
  <entity name="Card" pks="Card ID">
    <recobject-type nc="name" cardid="39509"/>
    <attribute name="Card ID"/>
    <expression transformations="Card Transformation">
      <tsource-attribute entity="BSCTarget" name="FQN"/>
      <between majority="minor">
        <min>0</min>
        <max>1000</max>
      </between>
      <regexp-expression name="RegExp Validation Rule" majority="major">
        <expression>^[[:digit:]]+$</expression>
      </regexp-expression>
      <reset attr name="CARD_ID" attrid="39509"/>
    </attribute>
    <attribute name="Parent ID" relation="Card to Device">
      <expression transformations="Card Transformation">
        <tsource-attribute entity="BSCTarget" name="Device"/>
        <reset attr name="PARENT_ID" attrid="39509"/>
      </expression>
    </attribute>
    <attribute name="Name" type="string">
      <expression transformations="Card Transformation" merge-rule="Card._/" alias="F_SN">
        <tsource-attribute entity="BSCTarget" name="Serial Number" alias="F_SN"/>
        <reset attr name="NAME" attrid="123456"/>
      </expression>
    </attribute>
  </entity>
</entity-set>
<entity name="Device" pks="Device ID">
  <recobject-type nc="name" cardid="39508"/>
  <attribute name="Name"/>
  <expression transformations="Device Transformation">
    <tsource-attribute entity="BSCTarget" name="Name"/>
    <vlike name="Like Validation Rule" majority="major">
      <expression>\w+\w+\w+</expression>
    </vlike>
  </attribute>
  <attribute name="Device ID" relation="Card to Device">
    <expression transformations="Device Transformation">
    </expression>
  </attribute>
</entity>

Exemplary Reconciled Data

NC_OBJECT_TYPES

<table>
<thead>
<tr>
<th>OBJECT_TYPE_ID</th>
<th>NAME</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>39508</td>
</tr>
<tr>
<td>2</td>
<td>39509</td>
</tr>
</tbody>
</table>

NC_ATTRIBUTES

<table>
<thead>
<tr>
<th>ATTR_ID</th>
<th>NAME</th>
<th>VALUE</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>39505</td>
<td>DEVICE_ID</td>
</tr>
<tr>
<td>2</td>
<td>39506</td>
<td>PARENT_ID</td>
</tr>
<tr>
<td>3</td>
<td>39507</td>
<td>CARD_ID</td>
</tr>
</tbody>
</table>

NC_OBJECTS

<table>
<thead>
<tr>
<th>OBJECT_ID</th>
<th>NAME</th>
<th>OBJECT_TYPE_ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Card_77221-77228</td>
<td>39509</td>
</tr>
<tr>
<td>2</td>
<td>Card_77223-77226</td>
<td>39509</td>
</tr>
<tr>
<td>3</td>
<td>Card_77224-77225</td>
<td>39509</td>
</tr>
<tr>
<td>4</td>
<td>Dev_1</td>
<td>39508</td>
</tr>
<tr>
<td>5</td>
<td>Dev_2</td>
<td>39508</td>
</tr>
<tr>
<td>6</td>
<td>Dev_3</td>
<td>39508</td>
</tr>
<tr>
<td>7</td>
<td>Dev_4</td>
<td>39508</td>
</tr>
</tbody>
</table>

NC_PARAMS

<table>
<thead>
<tr>
<th>ATTR_ID</th>
<th>OBJECT_ID</th>
<th>VALUE</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>39505</td>
<td>39513                                                             1 Value of attribute Device ID of device with Name “Dev_1”</td>
</tr>
<tr>
<td>2</td>
<td>39505</td>
<td>39514                                                             2 Value of attribute Device ID of device with Name “Dev_2”</td>
</tr>
<tr>
<td>3</td>
<td>39505</td>
<td>39515                                                             3 Value of attribute Device ID of device with Name “Dev_3”</td>
</tr>
<tr>
<td>4</td>
<td>39505</td>
<td>39516                                                             4 Value of attribute Device ID of device with Name “Dev_4”</td>
</tr>
<tr>
<td>5</td>
<td>39506</td>
<td>39510                                                             1 Value of attribute Parent ID of card with Name “Card_77221-77228”</td>
</tr>
<tr>
<td>ATTR_ID</td>
<td>OBJECT_ID</td>
<td>VALUE</td>
</tr>
<tr>
<td>---------</td>
<td>-----------</td>
<td>-------</td>
</tr>
<tr>
<td>6</td>
<td>39506</td>
<td>39511</td>
</tr>
<tr>
<td>7</td>
<td>39506</td>
<td>39512</td>
</tr>
<tr>
<td>8</td>
<td>39507</td>
<td>39510</td>
</tr>
<tr>
<td>9</td>
<td>39507</td>
<td>39511</td>
</tr>
<tr>
<td>10</td>
<td>39507</td>
<td>39512</td>
</tr>
</tbody>
</table>

[0161] The above-described systems and methods can be implemented in digital electronic circuitry, in computer hardware, firmware, and/or software. The implementation can be as a computer program product (i.e., a computer program tangibly embodied in an information carrier). The implementation can, for example, be in a machine-readable storage device, for execution by, or to control the operation of, a data processing apparatus. The implementation can, for example, be a programmable processor, a computer, and/or multiple computers.

[0162] A computer program can be written in any form of programming language, including compiled and/or interpreted languages, and the computer program can be deployed in any form, including as a stand-alone program or as a subroutine, element, and/or other unit suitable for use in a computing environment. A computer program can be deployed to be executed on one computer or on multiple computers at one site.

[0163] Method steps can be performed by one or more programmable processors executing a computer program to perform functions of the invention by operating on input data and generating output. Method steps can also be performed by an apparatus can be implemented as special purpose logic circuitry. The circuitry can, for example, be a FPGA (field programmable gate array) and/or an ASIC (application-specific integrated circuit). Modules, subroutines, and software agents can refer to portions of the computer program, the processor, the special circuitry, software, and/or hardware that implements that functionality.

[0164] Processors suitable for the execution of a computer program include, by way of example, both general and special purpose microprocessors, and any one or more processors of any kind of digital computer. Generally, a processor receives instructions and data from a read-only memory or a random access memory or both. The essential elements of a computer are a processor for executing instructions and one or more memory devices for storing instructions and data. Generally, a computer can include, can be operatively coupled to receive data from and/or transfer data to one or more mass storage devices for storing data (e.g., magnetic, magneto-optical disks, or optical disks).

[0165] Data transmission and instructions can also occur over a communications network. Information carriers suitable for embodying computer program instructions and data include all forms of non-volatile memory, including by way of example semiconductor memory devices. The information carriers can, for example, be EPROM, EEPROM, flash memory devices, magnetic disks, internal hard disks, removable disks, magneto-optical disks, CD-ROM, and/or DVD-ROM disks. The processor and the memory can be supplemented by, and/or incorporated in special purpose logic circuitry.

[0166] To provide for interaction with a user, the above described techniques can be implemented on a computer having a display device. The display device can, for example, be a cathode ray tube (CRT) and/or a liquid crystal display (LCD) monitor. The interaction with a user can, for example, be a display of information to the user and a keyboard and a pointing device (e.g., a mouse or a trackball) by which the user can provide input to the computer (e.g., interact with a user interface element). Other kinds of devices can be used to provide for interaction with a user. Other devices can, for example, be feedback provided to the user in any form of sensory feedback (e.g., visual feedback, auditory feedback, or tactile feedback). Input from the user can, for example, be received in any form, including acoustic, speech, and/or tactile input.

[0167] Any of the systems described herein can include clients and servers. A client and a server are generally remote from each other and typically interact through a communication network. The relationship of client and server arises by virtue of computer programs running on the respective computers and having a client-server relationship to each other.

[0168] The above described techniques can be implemented in a distributed computing system that includes a back-end component. The back-end component can, for example, be a data server, a middleware component, and/or an application server. The above described techniques can be implemented in a distributed computing system that includes a front-end component. The front-end component can, for example, be a client computer having a graphical user interface, a Web browser through which a user can interact with an example implementation, and/or other graphical user interfaces for a transmitting device. The components of the system can be interconnected by any form or medium of digital data communication (e.g., a communication network). Examples of communication networks include a local area network (LAN), a wide area network (WAN), the Internet, wired networks, and/or wireless networks.

[0169] Packet-based networks can include, for example, the Internet, a carrier internet protocol (IP) network (e.g., local area network (LAN), wide area network (WAN), campus area network (CAN), metropolitan area network (MAN), home area network (HAN)), a private IP network, an IP private branch exchange (IPBX), a wireless network (e.g., radio access network (RAN), 802.11 network, 802.16 network, general packet radio service (GPRS) network, Hiper-LAN), and/or other packet-based networks. Circuit-based
networks can include, for example, the public switched telephone network (PSTN), a private branch exchange (PBX), a wireless network (e.g., RAN, Bluetooth, code-division multiple access (CDMA) network, time division multiple access (TDMA) network, global system for mobile communications (GSM) network), and/or other circuit-based networks.

[0170] The network device can include, for example, a computer, a computer with a browser device, a telephone, an IP phone, a mobile device (e.g., cellular phone, personal digital assistant (PDA) device, laptop computer, electronic mail device), and/or other communication devices. The browser device includes, for example, a computer (e.g., desktop computer, laptop computer) with a world wide web browser (e.g., Microsoft® Internet Explorer® available from Microsoft Corporation, Mozilla® Firefox available from Mozilla Corporation). The mobile computing device includes, for example, a personal digital assistant (PDA).

[0171] Comprise, include, and/or plural forms of each are open ended and include the listed parts and can include additional parts that are not listed. And/or is open ended and includes one or more of the listed parts and combinations of the listed parts.

[0172] One skilled in the art will realize the invention may be embodied in other specific forms without departing from the spirit or essential characteristics thereof. The foregoing embodiments are therefore to be considered in all respects illustrative rather than limiting of the invention described herein. Scope of the invention is thus indicated by the appended claims, rather than by the foregoing description, and all changes that come within the meaning and range of equivalency of the claims are therefore intended to be embraced therein.

1. A computer implemented method for declarative and unified data transition, the method comprising:
   determining a unified configuration for a knowledge domain, the unified configuration comprising one or more predicates for one or more relationships between one or more system objects;
   generating one or more transformation rules based on the one or more predicates;
   transforming input data based on the one or more transformation rules, the input data comprising information associated with the one or more system objects, the one or more relationships between the one or more system objects, or any combination thereof;
   generating one or more reconciliation rules based on the one or more predicates, the one or more reconciliation rules associated with a destination database; and
   reconciling the transformed data with the destination database based on the one or more reconciliation rules.

2. The method of claim 1, further comprising:
   transforming second input data based on the one or more transformation rules, the second input data comprising information associated with the one or more system objects, the one or more relationships between the one or more system objects, or any combination thereof; and
   reconciling the second transformed data with the destination database based on the one or more reconciliation rules.

3. The method of claim 1, further comprising:
   generating one or more validation rules based on the one or more predicates; and
   validating the input data based on the one or more validation rules.

4. The method of claim 3, further comprising automatically modifying the input data based on the validation of the input data to remove at least one validation error associated with the input data.

5. The method of claim 1, further comprising extracting the input data from one or more source databases based on one or more extraction rules.

6. The method of claim 5, further comprising generating one or more extraction rules based on the one or more predicates.

7. The method of claim 5, wherein the extracting the input data from the one or more source databases and the reconciling the transformed data with the destination database occur relative to each other in real-time or substantially in real-time.

8. The method of claim 1, further comprising:
   generating one or more second reconciliation rules based on the one or more predicates, the one or more second reconciliation rules associated with a second destination database; and
   reconciling the transformed data with the second destination database based on the one or more second reconciliation rules.

9. The method of claim 1, wherein the transforming the input data further comprises joining at least one part of the input data with at least one other part of the input data based on the one or more transformation rules.

10. The method of claim 1, wherein the transforming the input data further comprises merging at least one part of the input data with at least one other part of the input data based on the one or more transformation rules.

11. The method of claim 1, further comprising:
   generating one or more post-transformation validation rules based on the one or more predicates; and
   validating the transformed data based on the one or more post-transformation validation rules.

12. The method of claim 1, further comprising merging the reconciled data with data stored on the destination database.

13. The method of claim 1, further comprising appending the reconciled data with data stored on the destination database.

14. The method of claim 1, wherein the input data comprises one or more table sets, one or more tables, one or more table fields, or any combination thereof.

15. The method of claim 1, wherein the destination database comprises an operational support system database, a healthcare system database, an information technology database, or any combination thereof.

16. The method of claim 1, further comprising:
   generating one or more second transformation rules based on the one or more predicates;
   transforming second input data based on the one or more second transformation rules, the second input data comprising information associated with the one or more system objects, the one or more relationships between the one or more system objects, or any combination thereof; and
   reconciling the first and second transformed input data comprising information associated with the one or more
system objects, the one or more relationships between the one or more system objects, or any combination thereof.

17. The method of claim 16, further comprising: generating one or more combined reconciliation rules based on the one or more predicates, the one or more combined reconciliation rules associated with one or more destination databases; and reconciling the first and second combined transformed input data with the one or more destination database based on the one or more combined reconciliation rules.

18. A computer implemented method for declarative and unified data transition, the method comprising: determining a unified configuration for a knowledge domain, the unified configuration comprising one or more predicates for one or more relationships between one or more system objects; generating one or more transformation rules based on the one or more predicates, the one or more transformations enabling transformation of input data, the input data comprising information associated with the one or more system objects, the one or more relationships between the one or more system objects, or any combination thereof; and generating one or more reconciliation rules based on the one or more predicates, the one or more reconciliation rules associated with a destination database and enabling reconciliation of the transformed data with the destination database.

19. A computer program product, tangibly embodied in an information carrier, the computer program product including instructions being operable to cause a data processing apparatus to:

determine a unified configuration for a knowledge domain, the unified configuration comprising one or more predicates for one or more relationships between one or more system objects;
generate one or more transformation rules based on the one or more predicates;
transform input data based on the one or more transformation rules, the input data comprising information associated with the one or more system objects, the one or more relationships between the one or more system objects, or any combination thereof;
generate one or more reconciliation rules based on the one or more predicates, the one or more reconciliation rules associated with a destination database; and reconcile the transformed data with the destination database based on the one or more reconciliation rules.

20. A system for declarative and unified data transition, the system comprising:
a unified configuration module configured to determine a unified configuration for knowledge domain, the unified configuration comprising one or more predicates for one or more relationships between one or more system objects;
a transformation rules engine configured to generate one or more transformation rules based on the one or more predicates;
a transformation module configured to transform input data based on the one or more transformation rules, the input data comprising information associated with the one or more system objects, the one or more relationships between the one or more system objects, or any combination thereof;
a reconciliation rules engine configured to generate one or more reconciliation rules based on the one or more predicates, the one or more reconciliation rules associated with a destination database; and
a reconciliation module configured to reconcile the transformed data with the destination database based on the one or more reconciliation rules.

21. The system of claim 20, further comprising:
the transformation module further configured to transform second input data based on the one or more transformation rules, the second input data comprising information associated with the one or more system objects, the one or more relationships between the one or more system objects, or any combination thereof; and
the reconciliation module further configured to reconcile the second transformed data with the destination database based on the one or more reconciliation rules.

22. The system of claim 20, further comprising:
a validation rules engine configured to generate one or more validation rules based on the one or more predicates; and
a validation module configured to validate the input data based on the one or more validation rules.

23. The system of claim 22, wherein the validation module is further configured to automatically modify the input data based on the validation of the input data to remove at least one validation error associated with the input data.

24. The system of claim 20, further comprising a source extraction module configured to extract the input data from one or more source databases based on one or more extraction rules.

25. The system of claim 24, further comprising a source extraction rule engine configured to generate the one or more extraction rules based on the one or more predicates.

26. The system of claim 20, wherein:
the reconciliation rules engine is further configured to generate one or more second reconciliation rules based on the one or more predicates, the one or more second reconciliation rules associated with a second destination database; and
the reconciliation module is further configured to reconcile the transformed data with the second destination database based on the one or more second reconciliation rules.

27. The system of claim 20, wherein the transformation module is further configured to join at least one part of the input data with at least one other part of the input data based on the one or more transformation rules.

28. The system of claim 20, wherein the transformation module is further configured to merge at least one part of the input data with at least one other part of the input data based on the one or more transformation rules.

29. The system of claim 20, wherein the transformation module is further configured to modify at least one part of the input data based on the one or more transformation rules.

30. The system of claim 20, further comprising:
a validation rules engine configured to generate one or more post-transformation validation rules based on the one or more predicates; and
a validation module configured to validate the transformed data based on the one or more post-transformation validation rules.
31. The system of claim 20, wherein the reconciliation module is further configured to merge the reconciled data with data stored on the destination database.

32. The system of claim 20, wherein the reconciliation module is further configured to append the reconciled data with data stored on the destination database.

33. The system of claim 20, wherein:

the transformation rules engine is further configured to generate one or more second transformation rules based on the one or more predicates;

the transformation module is further configured to transform second input data based on the one or more second transformation rules, the second input data comprising information associated with the one or more system objects, the one or more relationships between the one or more system objects, or any combination thereof;

the transformation rules engine is further configured to generate one or more combined transformation rules based on the one or more predicates; and

the transformation module is further configured to transform the first and second transformed input data based on the one or more combined transformation rules, the first and second transformed input data comprising information associated with the one or more system objects, the one or more relationships between the one or more system objects, or any combination thereof.

34. The system of claim 33, wherein:

the reconciliation rules engine is further configured to generate one or more combined reconciliation rules based on the one or more predicates, the one or more combined reconciliation rules associated with one or more destination databases; and

the reconciliation module is further configured to reconcile the first and second combined transformed input data with the one or more destination database on the one or more combined reconciliation rules.

35. A system for declarative and unified data transition, the system comprising:

a unified configuration module configured to determine a unified configuration for a knowledge domain, the unified configuration comprising one or more predicates for one or more relationships between one or more system objects;

a transformation rules engine configured to generate one or more transformation rules based on the one or more predicates, the one or more transformations enabling transformation of input data, the input data comprising information associated with the one or more system objects, the one or more relationships between the one or more system objects, or any combination thereof; and

a reconciliation rules engine configured to generate one or more reconciliation rules based on the one or more predicates, the one or more reconciliation rules associated with a destination database enabling reconciliation of the transformed data with the destination database.

36. A system for declarative and unified data transition, the system comprising:

means for determining a unified configuration for a knowledge domain, the unified configuration comprising one or more predicates for one or more relationships between one or more system objects;

means for generating one or more transformation rules based on the one or more predicates;

means for transforming input data based on the one or more transformation rules, the input data comprising information associated with the one or more system objects, the one or more relationships between the one or more system objects, or any combination thereof;

means for generating one or more reconciliation rules based on the one or more predicates, the one or more reconciliation rules associated with a destination database; and

means for reconciling the transformed data with the destination database on the one or more reconciliation rules.

37. A system for declarative and unified data transition, the system comprising:

means for determining a unified configuration for a knowledge domain, the unified configuration comprising one or more predicates for one or more relationships between one or more system objects;

means for generating one or more transformation rules based on the one or more predicates, the one or more transformations enabling transformation of input data, the input data comprising information associated with the one or more system objects, the one or more relationships between the one or more system objects, or any combination thereof; and

means for generating one or more reconciliation rules based on the one or more predicates, the one or more reconciliation rules associated with a destination database and enabling reconciliation of the transformed data with the destination database.

* * * * *