• Section 1 – Overview Features
• Section 2 – Introduction of KingHistorian
  - System structure
  - Interfaces supporting
• Section 3 – Collector usage
  - DCOM configuration
  - Collector usage
• Section 4 – KH Management Studio
• Section 5 – Comparison with others
• Section 1 – Overview Features
• Section 2 – Introduction of KingGraphic
  - Historical playback
  - Relational querying
  - 3rd-party support
• Section 3 – KingModel
• Section 4 – Application
  - KingHistorian, KingGraphic and KingModel combined together
• Section 1 – Overview Features
• Section 2 – Introduction of KingHistorian
  - System structure
  - Interfaces supporting
• Section 3 – Collector usage
  - DCOM configuration
  - Collector usage
• Section 4 – KH Management Studio
• Section 5 – Comparison with others
Real-Time Intelligent Platform (RTIP)

System Structure

Convert data into information
Data Collection

Massive data collection
No media restrictions
Buffering and resuming
Compressed data transmission
KingHistorian

Industrial historical database, can process massive industrial process data with concurrent storage and access, highly reliable, with high efficient performance and rich data access interfaces.

- Data storage and access,
- Robust and reliable,
- Save space, improve efficiency,
- Expandable data processing capability
Differences between Industrial Database and RDB

Industrial database deals with continuous industrial process data with timestamps.

Relational database deals with business data that have entity relationship structure.
Rich Open Interfaces

KingHistorian provides:

1. Rich data access interface, such as API, ODBC, OLEDB (ADO), SDK, etc.
3. SQL interface that conform to SQL-92 standard, and support extended SQL statement.
4. JAVA interface that support platform independent data access and operation.
Unique Data Compression Technology

A: Stored Value

Time

Measured Value

2010-11-18
Data Redundancy and Mirroring

Diagram showing the components of a network system including Hosts, RAID, Firewall, Redundancy, and Office/Data Network.
## KingHistorian vs INSQL

### Comparison Result for 70,000 Points

<table>
<thead>
<tr>
<th>Item</th>
<th>Type and Description</th>
<th>Amount</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPU</td>
<td>Duo core Intel 3GHz CPU</td>
<td>2</td>
</tr>
<tr>
<td>Memory</td>
<td>8G DDR2</td>
<td>1</td>
</tr>
<tr>
<td>Network card</td>
<td>1000M Network card (IP:172.16.2.210)</td>
<td>1</td>
</tr>
<tr>
<td>OS(32Bit)</td>
<td>Microsoft Windows Server 2003</td>
<td>1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>KingHistorian</th>
<th>INSQL</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPU Occupied (%)</td>
<td>1 Client: 3%</td>
<td>1 Client: 25%</td>
</tr>
<tr>
<td></td>
<td>10 Clients: 12%</td>
<td>10 Clients: 63%</td>
</tr>
<tr>
<td></td>
<td>15 Clients: 15%</td>
<td>15 Clients: 80%</td>
</tr>
<tr>
<td>Memory Used (G)</td>
<td>1 Client: 6G</td>
<td>1 Client: 1.93G</td>
</tr>
<tr>
<td></td>
<td>10 Clients: 6G</td>
<td>10 Clients: 1.97G</td>
</tr>
<tr>
<td></td>
<td>15 Clients: 6G</td>
<td>15 Clients: 2.0G</td>
</tr>
<tr>
<td>Network Occupied (M)</td>
<td>1 Client: 3M</td>
<td>1 Client: 2M</td>
</tr>
<tr>
<td></td>
<td>10 Clients: 8M</td>
<td>10 Clients: 2.8M</td>
</tr>
<tr>
<td></td>
<td>15 Clients: 9M</td>
<td>15 Clients: 3G</td>
</tr>
<tr>
<td>Query Speed</td>
<td>1 Client: 58000 records/s</td>
<td>1 Client: 20000 records/s</td>
</tr>
<tr>
<td></td>
<td>10 Clients: 5800 records/s</td>
<td>10 Clients: 3000 records/s</td>
</tr>
<tr>
<td></td>
<td>15 Clients: 5500 records/s</td>
<td>15 Clients: 2000 records/s</td>
</tr>
</tbody>
</table>
## KingHistorian & INSQL:

### KingHistorian3.0 VS PI 3.2.4.0

**Comparison result for 150,000 points**

<table>
<thead>
<tr>
<th></th>
<th>KingHistorian</th>
<th>PI</th>
</tr>
</thead>
<tbody>
<tr>
<td>Time of fully start up</td>
<td>41 s</td>
<td>2 min 22 s</td>
</tr>
<tr>
<td>Time of fully stop</td>
<td>10 s</td>
<td>1 min 30 s</td>
</tr>
<tr>
<td>File size of archive</td>
<td>19.9G</td>
<td>39G</td>
</tr>
<tr>
<td>Data</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Thread Amount**

<table>
<thead>
<tr>
<th></th>
<th>Amount</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>39</td>
</tr>
<tr>
<td>Piarchss</td>
<td>19</td>
</tr>
<tr>
<td>Pinetmgr</td>
<td>16</td>
</tr>
</tbody>
</table>

**CPU Occupied (%)**

<table>
<thead>
<tr>
<th></th>
<th>50 Client: 18%</th>
<th>100 Clients: 21%</th>
<th>120 Clients: 27%</th>
<th>150 Clients: 23%</th>
<th>200 Clients: 26%</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>50 Client: 10%</td>
<td>100 Clients: 10%</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>100 Clients: 10%</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>150 Clients: 23%</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>200 Clients: 26%</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Memory Used (G)**

<table>
<thead>
<tr>
<th></th>
<th>50 Client: 2.7G</th>
<th>100 Clients: 2.8G</th>
<th>120 Clients: 2.82G</th>
<th>150 Clients: 1.87G</th>
<th>200 Clients: 1.66G</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>50 Client: 1.1G</td>
<td>100 Clients: 1.44G</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>100 Clients: 1.44G</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>150 Clients: 1.87G</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>200 Clients: 1.66G</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Network Occupied (M)**

<table>
<thead>
<tr>
<th></th>
<th>50 Client: 200M</th>
<th>100 Clients: 420M</th>
<th>120 Clients: 420M</th>
<th>150 Clients: 1.25G</th>
<th>200 Clients: 1.4G</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>50 Client: 870M</td>
<td>100 Clients: 1.16G</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>100 Clients: 1.16G</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>150 Clients: 1.25G</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>200 Clients: 1.4G</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Query Speed**

<table>
<thead>
<tr>
<th></th>
<th>50 Client: 45000 records/s</th>
<th>100 Clients: 22000 records/s</th>
<th>120 Clients: 18000 records/s</th>
<th>150 Clients: 47 s/tag</th>
<th>200 Clients: 53 s/tag</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>50 Client: 27 s/tag</td>
<td>100 Clients: 37 s/tag</td>
<td>150 Clients: 47 s/tag</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Historical Server's configuration**

<table>
<thead>
<tr>
<th>Item</th>
<th>Type and description</th>
<th>Amount</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPU</td>
<td>Intel 3.2GHz CPU</td>
<td>4</td>
</tr>
<tr>
<td>Memory</td>
<td>3G DDR2</td>
<td>1</td>
</tr>
<tr>
<td>Network card</td>
<td>1000M Network card (IP:172.16.2.251)</td>
<td>1</td>
</tr>
<tr>
<td>OS(32 Bit)</td>
<td>Microsoft Windows Server 2003</td>
<td>1</td>
</tr>
</tbody>
</table>
DCOM Configuration:

1 Before using the OPC Collector, DCOM should be configured firstly.

1> use command in run-> Open with “dcomcnfg”
DCOM Configuration:

1. Before using the OPC Collector, DCOM should be configured firstly.
2. `<Component Services>`-> `<Computer>`-> `<My computer>`:
DCOM Configuration:

1. Before using the OPC Collector, DCOM should be configured firstly.

3. Select <DCOM> setting and select <OpcEnum> in the list box shown as below:
DCOM Configuration:

1 Before using the OPC Collector, DCOM should be configured firstly.

4> Select <OPCEnum> setting:

- Set <Start and activate authority> and add <everyone> user. Set the authority as <Local Start>, <Remote Start>, <Local Activate> and <Remote Activate>.
- Set <Visit Authority> and add <everyone> user. Set authority as <Local Visit> and <Remote Visit>.
- Set <Configuration Authority> and add <everyone> user. Set authority as <Whole Control> and <Read>
DCOM Configuration:

1. Before using the OPC Collector, DCOM should be configured firstly.

5. Select <Identify> setting:

   - Set <Properties> and then in <Identify> property page select <Interactive User>
OPC Collector Configuration:

1. Open “OpcCollector” configuration interface;
2. “New” to create a new collector;
3. Configure the collector settings;
4. Start the collector.
OPC Collector Configuration:

1. Open “OpcCollector” configuration interface;
2. “New” to create a new collector;
3. Configure the collector settings;
4. Start the collector.
OPC Collector Configuration:

Modification of collector properties: Select the collector and configure its properties.
KingHistorian: (KDBSysMgtStudio)

Start KingHistorian.
KingHistorian: Tag Manager
KingHistorian: Tag Manager

Import Tag:

Note: The tags in kingSCADA should be configure “Accessed by Other App”, so that in KingHistorian they can be accessed.
KingHistorian: Tag Manager

Configure tags properties:
KingHistorian: Tag Manager

Configure tags properties:

1 <General> Property Page:

- **Name**: It is the only logo of the data collector that connected to KingHistorian, collector in on KingHistorian project can not have same name.

- **Tag ID**: tag identifier

- **Tag Engineer Unit**: used to input engineer unit of the tag.

- **Description**: use to input the description information of the tag.
KingHistorian: Tag Manager

Configure tags properties:

2 <Collection> property page:
- Collect Data: Control the tag to collect data or not.
- Collector: Display all collector names in the list box.
- Source Address: To create correlation between database tag and collector tag which collector tag data will be collected.
- Data Type: KingHistorian provides 14 types tags.
- Data Length
- Collection Options: Polled, Solicited, Unknown
- Interval
- Offset
- Timestamp: Collector offers two main types of timestamp.
KingHistorian: Tag Manager

Configure tags properties:

3 <Archive and Compression> Property Page

- Save Historical Data
- Enable Multiple Version
- save server shutdown status
- Enable Collector Compress
- Compression mode of collector
- Timeout
- Dead-band
- Minimum Time Interval
- Storage Compression Mode
- Timeout
- Dead-band
- Minimum Time Interval

WellinTech Product Training — Yuehong Kan
**KingHistorian: RealTime Viewer**

View Real-time Data:

![Realtime Viewer](image)

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>OPC_Alarm.HighLimit</td>
<td>OPC Collector</td>
<td>df</td>
<td>0.000000</td>
<td></td>
<td>2009-09-27 15:24:52:843</td>
<td>0.000000</td>
<td>GOOD</td>
<td>0001</td>
</tr>
<tr>
<td>OPC_Alarm.HighStatus</td>
<td>OPC Collector</td>
<td>df</td>
<td>FALSE</td>
<td></td>
<td>2009-09-27 15:24:52:843</td>
<td>FALSE</td>
<td>GOOD</td>
<td>0001</td>
</tr>
<tr>
<td>OPC_Alarm.LowStatus</td>
<td>OPC Collector</td>
<td>df</td>
<td>FALSE</td>
<td></td>
<td>2009-09-27 15:24:52:843</td>
<td>FALSE</td>
<td>GOOD</td>
<td>0001</td>
</tr>
<tr>
<td>OPC_Alarm.MailStatus</td>
<td>OPC Collector</td>
<td>df</td>
<td>FALSE</td>
<td></td>
<td>2009-09-27 15:24:52:843</td>
<td>FALSE</td>
<td>GOOD</td>
<td>0001</td>
</tr>
</tbody>
</table>
KingHistorian: History Viewer

View Historical Data:

- Data Criteria
  - Note: By Time Range, By Number And Start Time, By Number And End Time

- History Data

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Value</th>
<th>Timestamp</th>
<th>Quality</th>
<th>Version</th>
</tr>
</thead>
<tbody>
<tr>
<td>OPC Collector TestOFCCollector 0FC_Local.dj1_rhvy</td>
<td>Int32</td>
<td>33</td>
<td>2010-09-22 16:05:12.643</td>
<td>GOOD</td>
<td>0</td>
</tr>
<tr>
<td>OPC Collector TestOFCCollector 0FC_Local.dj1_rhvy</td>
<td>Int32</td>
<td>47</td>
<td>2010-09-22 16:05:12.643</td>
<td>GOOD</td>
<td>0</td>
</tr>
<tr>
<td>OPC Collector TestOFCCollector 0FC_Local.dj1_rhvy</td>
<td>Int32</td>
<td>12</td>
<td>2010-09-22 16:05:10.627</td>
<td>GOOD</td>
<td>0</td>
</tr>
<tr>
<td>OPC Collector TestOFCCollector 0FC_Local.dj1_rhvy</td>
<td>Int32</td>
<td>38</td>
<td>2010-09-22 16:05:08.887</td>
<td>GOOD</td>
<td>0</td>
</tr>
<tr>
<td>OPC Collector TestOFCCollector 0FC_Local.dj1_rhvy</td>
<td>Int32</td>
<td>37</td>
<td>2010-09-22 16:05:00.012</td>
<td>GOOD</td>
<td>0</td>
</tr>
<tr>
<td>OPC Collector TestOFCCollector 0FC_Local.dj1_rhvy</td>
<td>Int32</td>
<td>51</td>
<td>2010-09-22 16:05:01.536</td>
<td>GOOD</td>
<td>0</td>
</tr>
<tr>
<td>OPC Collector TestOFCCollector 0FC_Local.dj1_rhvy</td>
<td>Int32</td>
<td>34</td>
<td>2010-09-22 16:05:06.586</td>
<td>GOOD</td>
<td>0</td>
</tr>
<tr>
<td>OPC Collector TestOFCCollector 0FC_Local.dj1_rhvy</td>
<td>Int32</td>
<td>3</td>
<td>2010-09-22 16:05:04.593</td>
<td>GOOD</td>
<td>0</td>
</tr>
<tr>
<td>OPC Collector TestOFCCollector 0FC_Local.dj1_rhvy</td>
<td>Int32</td>
<td>21</td>
<td>2010-09-22 16:05:03.555</td>
<td>GOOD</td>
<td>0</td>
</tr>
<tr>
<td>OPC Collector TestOFCCollector 0FC_Local.dj1_rhvy</td>
<td>Int32</td>
<td>11</td>
<td>2010-09-22 16:05:02.585</td>
<td>GOOD</td>
<td>0</td>
</tr>
</tbody>
</table>
KingHistorian: SQL Statement Query

View SQL Data:

- `select * from History;`
- `execute SQLStatement;`
- `set sqlstatement;`

2010-11-18
Practice & Discussion

1 About Collector?
2 Realtime viewer?
3 History viewer?
4 SQL Statement query?
• Section 1 – Overview Features
• Section 2 – Introduction of KingGraphic
  - Historical playback
  - Relational querying
  - 3\textsuperscript{rd}-party support
• Section 3 – KingModel
• Section 4 – Application
  - KingHistorian, KingGraphic and KingModel combined together
KingGraphic: to display data form database (relational database or historian).
KingGraphic:

Configure data source from KingHistorian:
Configure data source from KingModel:
Create animations on picture:
History playback: provides several functions to finish historical data playback.
KingGraphic:

History playback: using “hy.kgn” genius. (you can get the genius from WellinTech)

1 create a picture to put on the genius from lib “hy”
KingGraphic:

History playback: using “hy.kgn” genius. (you can get the genius from WellinTech)

2 create a picture to show the data from historian.

Note: if want to show the historical playback through Pointer object, assign object to the pointer object before the playback operation.
History playback: using “hy.kgn” genius. (you can get the genius from WellinTech).

3 Use historical genius to replay the picture. (Only input the picture name into the “PicName” textbox.)
KingGraphic:

History playback toolbar genius:

- Start to playback
- Roll back
- Forward
- Pause
- Linear interpolation
- step interpolation

- Fail to access the historian
- Realtime state
- Historical state

- Picture name to be playback
- Starttime of the playback operation
- Endtime of the playback operation
- Speed of the playback
- Process of the whole procedure
Array:

```cpp
void OnOpen()
{
    CreatArray("dj", 3, 1);
    SetArrayElement("dj", 0, StrFromInt(UCComboBox1.SelectIndex+1, 10));
}
```
Array:

```c
string temp;
temp=GetArrayElement("dj", 0);
KModelRef("KMSource1", "djPointer", "dj"+temp)
```

Introduction
1. About KingGraphic?
2. About playback?
3. About connection with KH?
• Section 1 – Overview Features
• Section 2 – Introduction of KingGraphic
  - Historical playback
  - Relational querying
  - 3rd-party support
• Section 3 – KingModel
• Section 4 – Application
  - KingHistorian, KingGraphic and KingModel combined together
**KingModel:**

**Background of the requirement of KingModel:**
- Switch picture base on “Object”
- View and query information of the same type of devices
- Configure once using only one picture (by using KingModel)

**Generally to say:**

KingModel can manage the tags in KingHistorian as groups.

Its model supports to be created, imported and exported in batch, and also provides industrial display graphical interface to show data information as objects.
KingModel: grouping the tags of KingHistorian using models.

LogIn KingModel:

- **Name**: name of the new data source, should be matched with a historian
- **UserName**: fill the blank with “sa”
- **Password**: default password is “sa”
- **Server**: Historian server address; 127.0.0.1 refer to local
- **Port**: Default set as 7890, can be modified by file “KMConfig.ini”, should restart model service after the modification.
- **Timeout**: the setting of timeout, unit: ms
KingModel: grouping the tags of KingHistorian using models.

LogIn KingModel:

- **Menu:**
- **Toolbar**
- **Tree:** the entrance of funcions
- **Content Display Area**
- **Output Window:** message
KingModel: grouping the tags of KingHistorian using models.

ToolBar:
- **Connect**: to connect with KingHistorian server
- **Disconnect**: to disconnect with KingHistorian server
- **New**: create new data source or model
- **Delete**: delete current selected item
- **Property**: pop up the property window

Output window:
- Show successful or failed information
- Show debugging information
- Show errors and warning information
KingModel: Create Model

Use “New” on the toolbar, and create new model.

1 General setting: name and description
2 Attribute setting:
   Right click the blank to create a new attribute; the attribute can make a correspondence relation between model variables and variables in historian.

Note:
Setting: to format the “Setting” can set wildcard characters which are %Object%.%Attribute%, it means when the model is instantiated to be objects, the properties of the objects will be in accordance with the “model name. attribute name”.
For example: assuming generating an object from model 1 and name it “object 1”, then the variable of properties 1 of the object 1 will automatically configured to connect the local database “Object 1.Property 1”, then this connection is subject to the local industrial base with “Object1.Property1”.
KingModel: Create Model

Use “New” on the toolbar, and create new model.

3 Parameter setting: To create new static parameter for the model, its value can be passed to model objects.

- Name:
- Description:
- DataType:
- DefaultValue:

After these configurations, a model setting is completed.

Objects generated from model:

- Objects can be instantiated by models, and they can inherit properties and parameters.
- Objects include common objects and pointer type objects, the only difference between them is pointer objects needn’t link to real tags.
KingModel: Create Model

Create objects and object management:

1 Object Group

Click “OBJECTGROUP->New Group”, Create a group.

For instance, the parameters of same type of motors can be divided into several groups, one group can have many objects. For example, a motor model has been created, and there are a number of motor scenes of the template has a number of electric motor, in order to facilitate the management, we can create a motor object group, and place the specific objects (dj1, dj2 ..., djN) in this group.

2 Create object or objects:
KingModel: Create Model

Create objects and object management:

2 Create object or objects:

1> New Object:

2> New Objects: to create objects in batch, Choose the Model you want to generate objects.

- Fill the number of names the user wants to create;
- To use (*) in the “Name” blank to generate names with same features, like “djzw1”, “djzw2”, …, “djzwN”;
- Click “Generate Name”, get new name list on the right side.
Practice & Discussion

1. About KingModel?

2. Q & A
• Section 1 – Overview Features
• Section 2 – Introduction of KingGraphic
  - Historical playback
  - Relational querying
  - 3\textsuperscript{rd}-party support
• Section 3 – KingModel
• Section 4 – Application
  - KingHistorian, KingGraphic and KingModel combined together
Application of using KingHitorian + KingGraphic + KingModel
Use KingModel in KingGraphic to display data from KingHistorian:

1. Prepare a KingSCADA or KingView project to simulate or generate real-time data for this example.

   1> Tags need to be created: (All tags can be memory data type or IO float, just to make sure it can generate data change.)

   - dj1_rhyw: float
   - dj1_djzw: float
   - dj1_rhyy: float
   - dj2_rhyw: float
   - dj2_djzw: float
   - dj2_rhyy: float
   - dj3_rhyw: float
   - dj3_djzw: float
   - dj3_rhyy: float

2> Insert three genius into a picture to show the tags' value. (Link ''Value'' with tags defined.)
Use KingModel in KingGraphic to display data from KingHistorian:

2 Configure in KingHistorian.

1> Create new OPC Collector to connect with KingSCADA or Kingview or other datasource. Use OPC Collector to connect with KingSCADA for instance.
Use KingModel in KingGraphic to display data from KingHistorian:

2 Configure in KingHistorian.

2> Create tag group in KingHistorian:

Collector Manager: to make sure the OPCCollect is running normally.

Create tag group “Motor”, and import tags from KingSCADA:

Import tags from Collector:
Use KingModel in KingGraphic to display data from KingHistorian:

2 Configure in KingHistorian.

2> Change “OPC_Local.” to “OPC_Local.dj”, for “.” is used to as special symbol in the script of KingGraphic and KingModel.

Export tag group to Excel, change column “TagName” using “Motor” to substitute “OPC_Local.dj.”
Use KingModel in KingGraphic to display data from KingHistorian:

2 Configure in KingModel.

1> Create new datasource

2> Create new model

DataSource Property

<table>
<thead>
<tr>
<th>Name:</th>
<th>KM1</th>
</tr>
</thead>
<tbody>
<tr>
<td>UserName:</td>
<td>z9</td>
</tr>
<tr>
<td>Password:</td>
<td>***</td>
</tr>
<tr>
<td>Server:</td>
<td>127.0.0.1</td>
</tr>
<tr>
<td>Port:</td>
<td>5678</td>
</tr>
<tr>
<td>Timeout:</td>
<td>1000</td>
</tr>
</tbody>
</table>

New Model Property

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
<th>Data Type</th>
<th>DataSource</th>
<th>Setting</th>
</tr>
</thead>
<tbody>
<tr>
<td>RHPy</td>
<td>RHPy</td>
<td>REAL32</td>
<td>KM</td>
<td>Motor1_chy</td>
</tr>
<tr>
<td>RHPv</td>
<td>RHPv</td>
<td>REAL32</td>
<td>KM</td>
<td>Motor1_chy</td>
</tr>
<tr>
<td>RHPw</td>
<td>RHPw</td>
<td>REAL32</td>
<td>KM</td>
<td>Motor1_chy</td>
</tr>
</tbody>
</table>
Use KingModel in KingGraphic to display data from KingHistorian:

2 Configure in KingModel.

3. Create new objects, Motor1, Motor2, Motor3, (Object type “Object”); MotorPointer (Object type “Pointer”)
Use KingModel in KingGraphic to display data from KingHistorian:

3 Create KingGraphic project.
   1> Create new project

![New project dialog box]

Name: Motor
Location: C:\Program Files\KingGraphic\My Projects\,
Solution: New solution
Solution name: Motor
Informations:
Author: DELL
Company:
Create time: 2010-09-23 07:53:30
Description: 

2010-11-18 WellinTech Product Training — Yuehong Kan 64 www.wellintech.com
Use KingModel in KingGraphic to display data from KingHistorian:

3 Create KingGraphic project.

2> Add Data Model
Use KingModel in KingGraphic to display data from KingHistorian:

3 Create KingGraphic project.
3> Create new picture

Motor1’s djzw: #
Motor1’s rhyy: #
Motor1’s rhyw: #

Motor2’s djzw: #
Motor2’s rhyy: #
Motor2’s rhyw: #

Motor3’s djzw: #
Motor3’s rhyy: #
Motor3’s rhyw: #

Pointer of Motor1’s djzw: #
Pointer of Motor1’s rhyy: #
Pointer of Motor1’s rhyw: #

2010-11-18
Use KingModel in KingGraphic to display data from KingHistorian:

3 Add Model Control.

3> KingModelObjectBrowse, KingModelObjectComboBrowser, KingModelObjectListBrowser

Add “KingModelObjectComboBrowser” through

```
Application
```

2010-11-18
Practice & Discussion

1. About all usage of these products?

2.
Thanks

www.wellintech.com