BCDSAgent (Ver. 2)
User's Guide

Masato Takeichi

2019/06/16

Added Features (Ver. 2.0-2.1)

Minor Improvement (Ver. 2.2)

New Features of Ver. 2.1 - Differences from Ver. 2.0

- Configuration File and SQL File for BData are unifiled into a single new Configuration File.
  - Agent configuration is described as 'comments' of SQLite statements, i.e., lines starting '--'.
  - Accordingly, BData and IPort specification should begin with '--#'. And a separate SQL files are no more used, use configuration files if necessary.
  - File extension is arbitrary, but '.sql' would be better for interactive SQL execution.
- SQL statements described in configuration files are executed at all times when BData is updated, whereas SQL execution may be suppressed and put it into user's control with the SQLite CLI Application or 'DB Browser for SQLite'.
  - To suppress SQL execution, a line beginning with '--$$' should be included in the configuration file.

Running BCDSAgent Ver. 2.1

- Rewriting Configuration Files for Ver. 2.1 from those of Ver. 2.0

Configuration File uBio.conf — Ver. 2.0

```
-- uBio
#BData uBio CDSS.db 55557 >G_n >G_c U_n>! U_c>!
#IPort GUS localhost 55556 >G_n >G_c
```

SQL File uBio.SQL

```
INSERT INTO uBio (U_n,U_c) SELECT G_n,G_c FROM uBio
```

Make sure to put a 'double dash" in front of #BData and #IPOrt lines.

Configuration File uBio.sql — Ver. 2.1

```
-- uBio
--#BData uBio CDSS.db 55557 >G_n >G_c U_n>! U_c>!
--#IPort GUS localhost 55556 >G_n >G_c
INSERT INTO uBio (U_n,U_c) SELECT G_n,G_c FROM uBio
```

- Including a line beginning with '--$$' in the Configuration File to invoke Interactive Database Application as in Ver. 2.0

```
-- uBio
--$$
--#BData uBio CDSS.db 55557 >G_n >G_c U_n>! U_c>!
--#IPort GUS localhost 55556 >G_n >G_c
INSERT INTO uBio (U_n,U_c) SELECT G_n,G_c FROM uBio
```

Note on Execution of Application

- SQL statements described in configuration files are used to implement the Agent's 'Update Policy' as an 'Application' under the 'Bidirectional View-Update' scheme.
- However, automatic execution of the Application may fall into an infinite 'oscillation' or 'propagation' of view-update.
  - Specification of columns of the BCDS View Table with 'ReadOnly' ('>') and 'ReadWrite' ('>!') may solve these problems between Agents in some cases. There remain more subtle cases, however, in construction of BCDS.
  - Interactive invocation mode is provided to find out such situations and to seek some mechanisms for effective BCDS.
- See Next Page for a strategy for avoiding the 'Infinite' in Ver. 2.1

Strategies for Avoiding the 'Infinite' in Ver. 2.1

- In BCDSAgent Ver. 2.1, following strategy has been implemented for avoiding infinite oscillations and propagations.
  - Check the changes taken by Application by comparing BData before execution and after execution of Application, and by update propagation through IPorts and OPorts.
  - If no changes in BData are observed, no propagation is taken; Otherwise, they are propagated to IPorts and OPorts, and to other Agents through these Ports.
- Comparing BData at every time when BData is updated through IPorts and OPorts, and Application is rather costly and should be improved. It might be better to integrate this problem with 'Incrementalization' of view-updating information.

BCDSAgent (Ver. 2.0)
User's Guide

Masato Takeichi

2019/06/13

Revised 2019/06/14

(Pages for Windows OS Added)

Using BCDSAgent (Ver.2) on Mac OS X

Unzip the file 'BCDSAgent.zip' at any place on your home directory before proceeding to next step.

Contents distributed:

- 'BCDSAgent (Ver.2) Users Guide' (this file) explains how to use this 'Bidirectional Collaborative Data Sharing' (BCDS) for experiments.
- 'BCDSAgent-exe' : Binary executable code
- 'Orchestra' contains configuration files and database for demonstration; this simulate an example of CDSS from 'Orchestra' Paper of ACM TODS 38(3), 2013
- 'DejimaXYZ' contains configuration files and database for constructing Dejima with three Sites.

'BCDSAgent (Ver.2)' was developed under Mac OS X 10.14.5, and was tested on Mac OS X 10.12, too. There will be no problems running on recent versions of Mac OS.

Using BCDSAgent (Ver.2) on Windows 10

Unzip the file 'BCDSAgent-win.zip' at any place on your home directory before proceeding to next step.

Contents distributed:

- 'BCDSAgent (Ver.2) Users Guide' (this file) explains how to use this 'Bidirectional Collaborative Data Sharing' (BCDS) for experiments.
- 'BCDSAgent-exe.exe' : Binary executable code
- 'sqlite3.dll' : Dynamic library used at run time.
- 'Orchestra' contains configuration files and database for demonstration; this simulate an example of CDSS from 'Orchestra' Paper of ACM TODS 38(3), 2013
- 'DejimaXYZ' contains configuration files and database for constructing Dejima with three Sites.

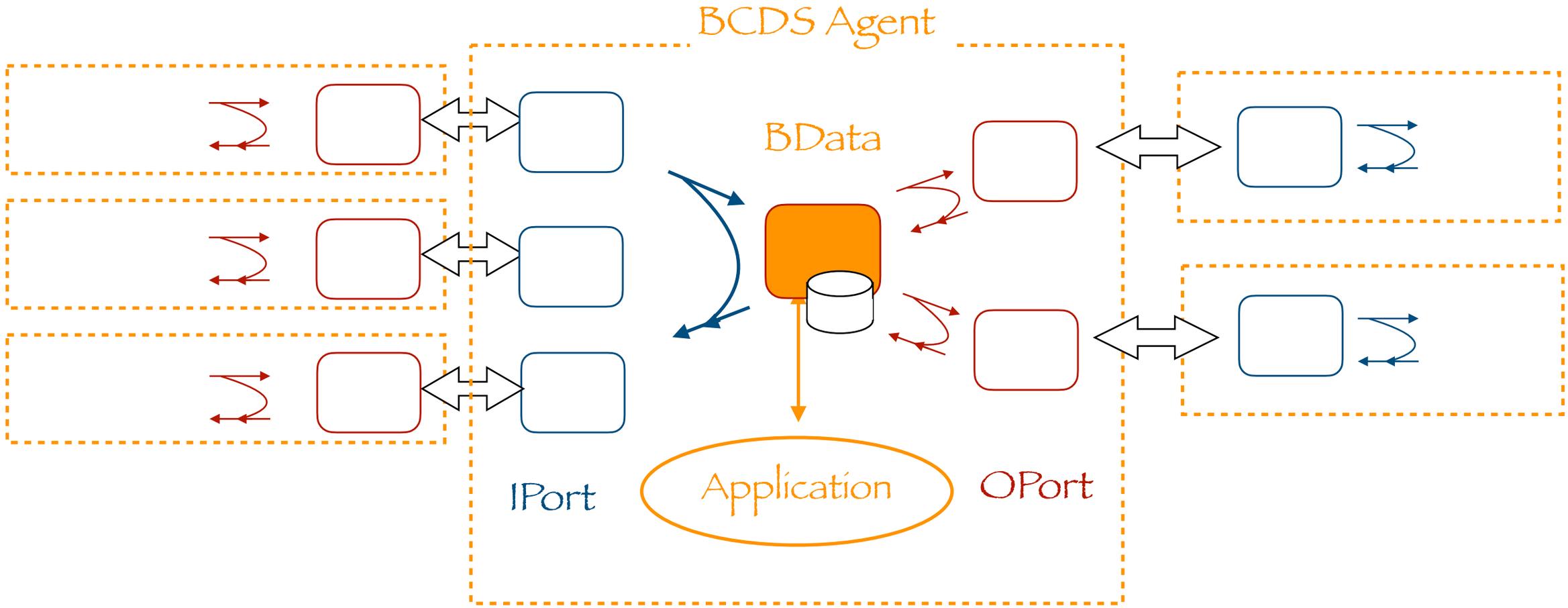
Some Notes for Windows Users

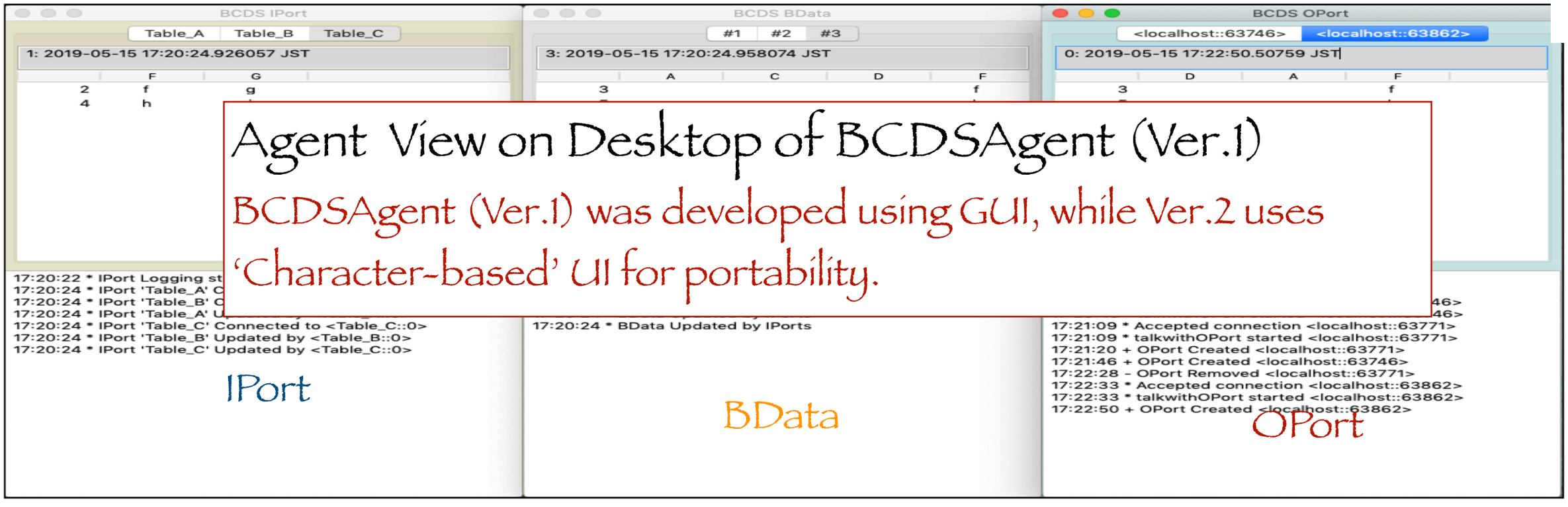
- This document assumes that
  - Our working directory is named 'BCDSAgent'; rename the folder name if necessary.
  - The file path is written in the form 'a/b/...' instead of Windows notation 'a\b\...'.
  - We will use 'Terminal' for shell commands, which corresponds to the 'Command Prompt' application of Windows.
- A dynamic library 'sqlite3.dll' was not supplied from standard archives of Haskell library, and it is placed at 'BCDSAgent' directory with BCDSAgent-exe executable. If you like to move executable code to other places, make sure to be able to access this DLL at runtime.
- For SQLite Database on Windows, we need to install 'DB Browser for SQLite' only; No need to install Command Line Shell application for SQLite.

What we need for experimentation of BCDS

- We are strongly advised to use 'DB Browser for SQLite' to prepare persistent data (database table) of participant Sites of BCDS Agent and to update the View of their own data.
  - We may use 'sqlite3' command line application provided by Mac OS to do these tasks, which might be tedious work to do with 'sqlite3' commands.
  - In practical situation, we need to code programs which watch Sites' base data and are activated every time upon the base data updated.
  - We are now trying to find out what should be done by these applications using BCDSAgent (Ver.2). BCDSAgent helps users watch the status of our collaboration Sites and how 'Bidirectional' view-update propagates.
  - In our next step to configure BCDS is to write proper applications based on our experience.
- 'DB Browser for SQLite' can be downloaded from
  https://sqlitebrowser.org

Schematic View of BCDS Agent Showing Names of its Components



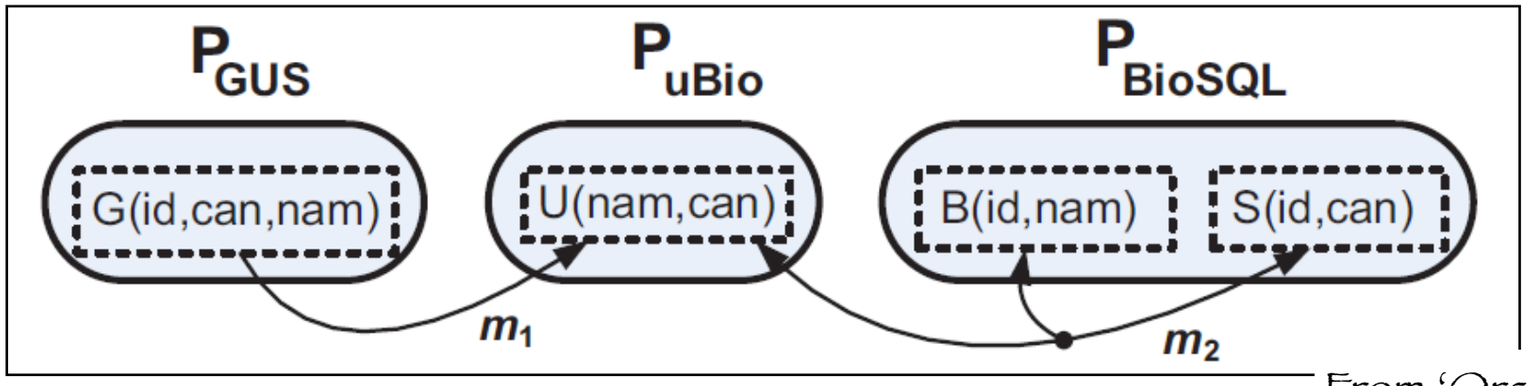


Agent View on Desktop of BCDSAgent (Ver.1)

BCDSAgent (Ver.1) was developed using GUI, while Ver.2 uses 'Character-based' UI for portability.

Running BCDS 'Orchestra' Example Step by Step

Follow instructions after the overview pages of the example.



$(m_1) \quad G(i,c,n) \rightarrow U(n,c)$

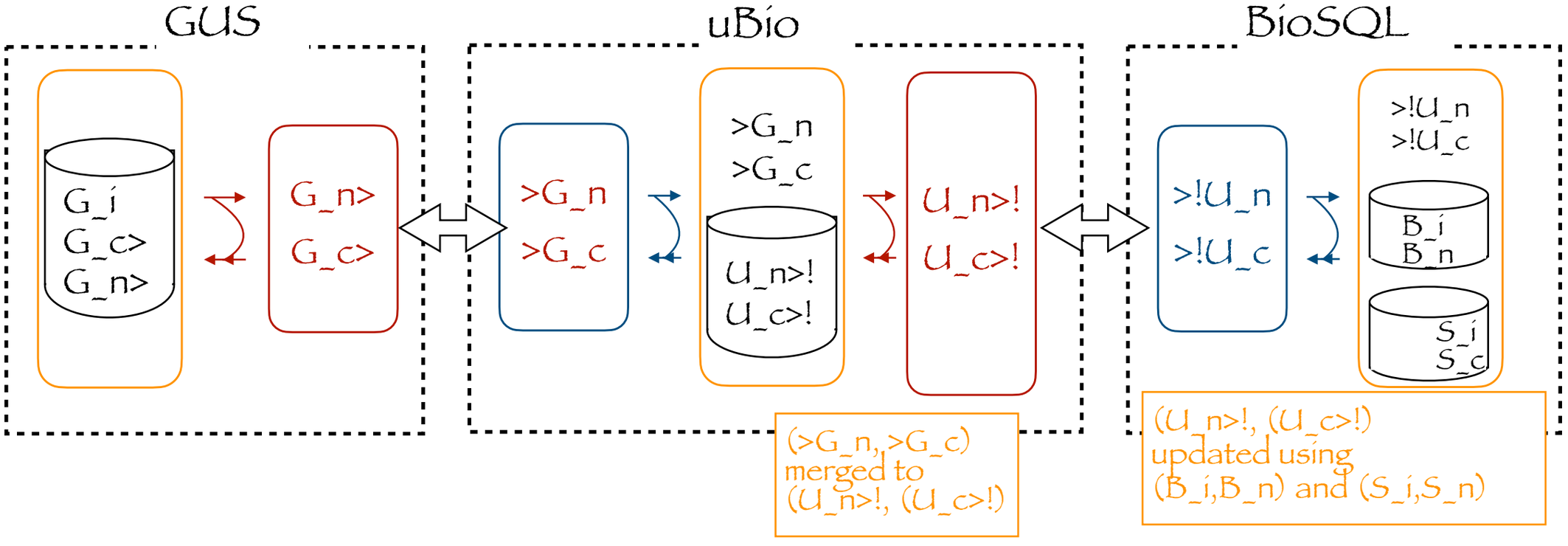
$(m_2) \quad \exists i \; B(i,n) \wedge S(i,c) \leftrightarrow U(n,c)$

From 'Orchestra' Paper of ACM TODS 38(3), 2013

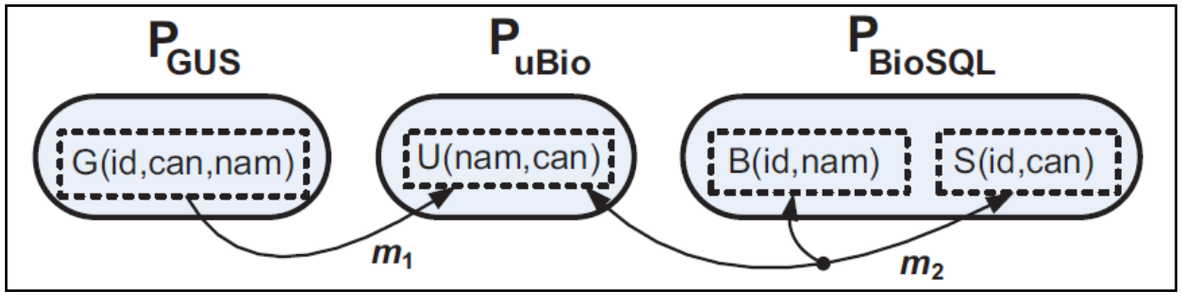
-- GUS
#BData GUS CDSS.db 55556 -
G_i G_c> G_n>

-- uBio
#BData uBio CDSS.db 55557 -
>G_n >G_c U_n>! U_c>!
#IPort GUS localhost 55556 -
>G_n >G_c

-- BioSQL
#BData BioSQL CDSS.db 55558-
>!U_n >!U_c
#IPort uBio localhost 55557 -
>!U_n >!U_c



—query $ans(x, y)$:- $B(x, y)$ returns $\{(3, 5)\}$;
—query $ans(x, z)$:- $B(y, x), S(y, z)$ returns $\{(2, 5), (3, 2), (5, 4)\}$



G

1	4	5
3	5	2

B

3	5
c_1	2
c_2	3
c_3	5

U

3	2
2	5
5	4

S

3	4
c_1	5
c_2	2
c_3	4

To be producedfrom G, B and S

Database Structure | Browse Data | Edit Pragmas | Execute SQL

Table: GUS

	G_i	G_c	G_n
	Filter	Filter	Filter
1	1	4	5
2	3	5	2

Database Structure | Browse Data | Edit Pragmas | Execute SQL

Table: BioSQL_B

	B_i	B_n
	Filter	Filter
1	3	5
2	101	2
3	102	3
4	103	5

Database Structure | Browse Data | Edit

Table: BioSQL_S

	S_i	S_c
	Filter	Filter
1	3	4
2	101	5
		2
		4

101, 102, and 103 for c_1, c_2, and c_3

SQL code for uBio

```
INSERT INTO uBio (U_n,U_c)
    SELECT G_n,G_c FROM uBio;
```

```
CREATE VIEW BioSQL_V AS
SELECT * FROM BioSQL_B
  INNER JOIN BioSQL_S ON B_i=S_i;
INSERT INTO BioSQL
 SELECT B_n, S_c FROM BioSQL_V
 WHERE NOT EXISTS
 (SELECT U_n, U_c FROM BioSQL
  WHERE U_n=B_n AND U_c=S_c);
DROP VIEW BioSQL_V
```

SQL code for BioSQL

$(m_2^{\rightarrow})$ $B(i, n) \wedge S(i, c) \rightarrow U(n, c)$

$(m_2^{\leftarrow})$ $U(n, c) \rightarrow \exists i\ B(i, n) \wedge S(i, c)$

Making no duplicate insertion in BioSQL Table

Snapshot of BCDSAgent Running on Mac OS

Orchestra — BCDSAgent-exe GUS.conf — 80×31

```
Laphroaig:BCDSAgent takeichi$ cd Orchestra
Laphroaig:Orchestra takeichi$ ../BCDSAgent-exe GUS.conf
* BCDS Agent Configuration Specification
  -- GUS
  #BData GUS CDSS.db 55556 G_i G_c> G_n>
* BCDS Agent Configuration Successful
2019-06-14 10:51:47.06881 JST * BData Database [CDSS.db::GUS] Opened
* BCDS Agent Initialization Completed
* BCDS Agent Command Line Interface Started
2019-06-14 10:51:47.068949 JST * BData Service Started
- BCDSAgent: 2019-06-14 10:51:47.075194 JST * Listening on Port 55556

    '*'  for Drawing BData
    '>*' for Drawing IPort
    '*>' for Drawing OPort
    '>>*' for Reconnecting IPort
    '>$' for Transferring BData to Application
    '$>' for Updating BData from Application
- BCDSAgent: *
BData 'GUS' #0:2019-06-14 10:51:47.068499 JST

____|___G_i___|__G_c>___|__G_n>___
1   |1        |4        |5
2   |3        |5        |2

- BCDSAgent: 2019-06-14 10:52:42.961519 JST * ta
ted at 58252
2019-06-14 10:52:42.961755 JST + OPort 'GUS' Cre
2019-06-14 10:52:42.962069 JST * OPort 'GUS' Sen
2019-06-14 10:52:42.962731 JST * OPort 'GUS' Sen
- BCDSAgent:
```

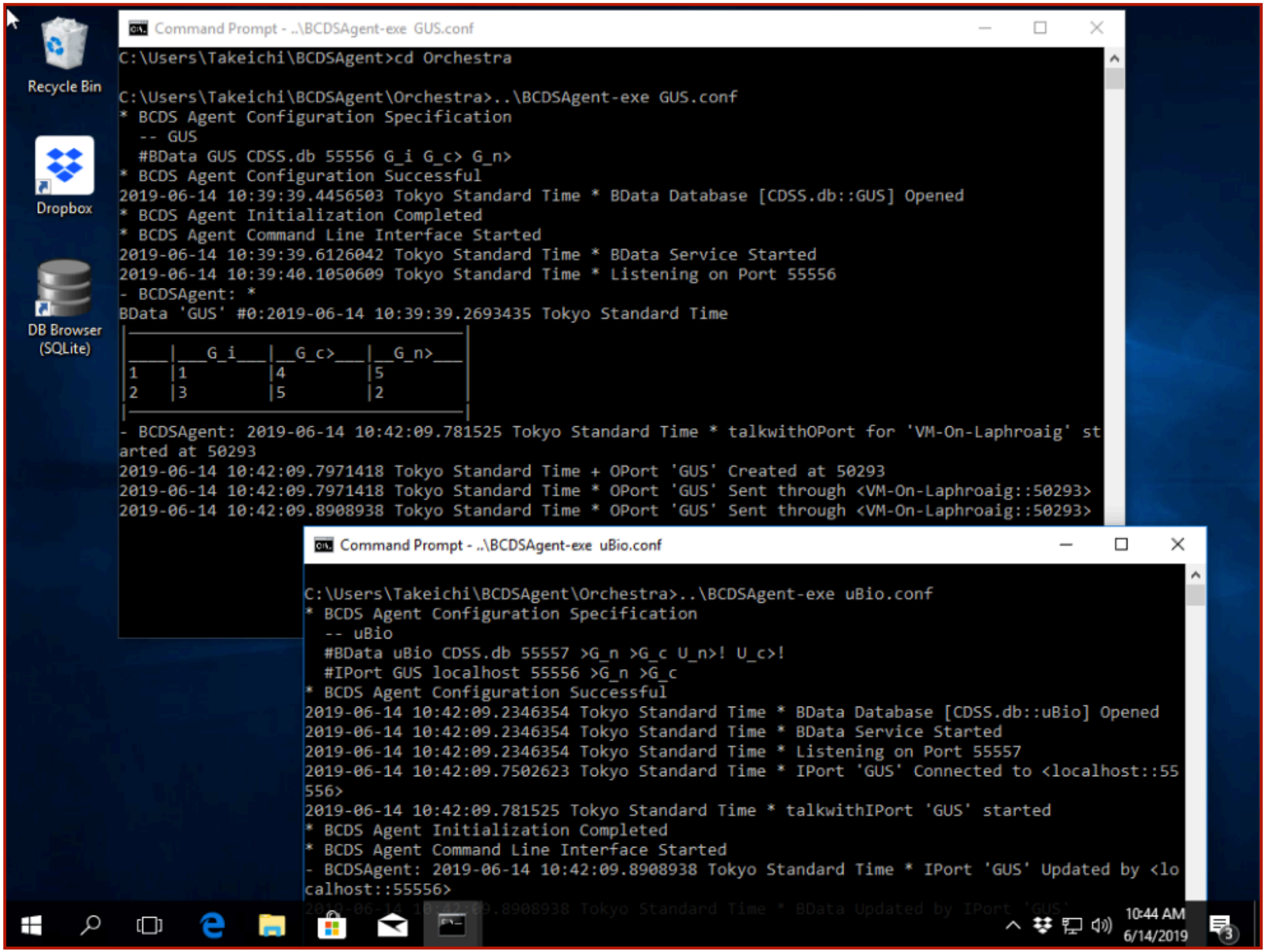
Orchestra — BCDSAgent-exe uBio.conf — 70×24

```
  #IPort GUS localhost 55556 >G_n >G_c
* BCDS Agent Configuration Successful
2019-06-14 10:52:42.95694 JST * BData Database [CDSS.db::uBio] Opened
2019-06-14 10:52:42.957323 JST * BData Service Started
2019-06-14 10:52:42.959488 JST * Listening on Port 55557
2019-06-14 10:52:42.960484 JST * IPort 'GUS' Connected to <localhost::
55556>
2019-06-14 10:52:42.960819 JST * talkwithIPort 'GUS' started
* BCDS Agent Initialization Completed
* BCDS Agent Command Line Interface Started
2019-06-14 10:52:42.963167 JST * IPort 'GUS' Updated by <localhost::55
556>
2019-06-14 10:52:42.963794 JST * BData Updated by IPort 'GUS'
- BCDSAgent: *
BData 'uBio' #1:2019-06-14 10:52:42.963653 JST

____|__>G_n___|__>G_c___|__U_n>!__|__U_c>!__
2   |5        |4        |  -----  |  -----
3   |2        |5        |  -----  |  -----
5   |  -----  |  -----  |2        |5
7   |5        |4        |  -----  |  -----
8   |2        |5        |  -----  |  -----

- BCDSAgent:
```

Snapshot of BCDSAgent Running on Windows 10



Step 1: Opening 'Terminal' and 'DB Browser for SQLite'

1. Open 'Terminal' and move to 'BCDSAgent/Orchestra' folder. Do shell command '$ ls' to see the files contained in this folder.
   - 'CDSS.db' database file containing tables used in this collaboration system.
   - Three '*.conf' files for Agent Configuration Specification of the Sites.
   - Two '*.SQL' files used for View-updating using SQLite Database System.
2. Open 'DB Browser for SQLite': Click 'Open Database' button and locate 'BCDSAgent/Orchestra' to open 'CDSS.db'. We can find 5 tables listed.
   - Click 'Browser Data' tab for viewing the table contents. We can find that 'BioSQL' and 'uBio' contain nothing , 'BioSQL_B' and 'BioSQL_S' contain 4 records each, and 'GUS' contains 2 records.
   - Do not need to close the Browser. But be careful not to forget press 'Write Changes' after editing the View. Otherwise the database is locked and other program including our Agent cannot access the contents.

Step 2: Starting 'Site_uBio'

3. Return to the 'Terminal' and do '$ ../BCDSAgent-exe uBio.conf'.
   - This starts 'Site_uBio'. We see several messages from the Agent:
     - Copy of Configuration Specification given by 'uBio.conf'
     - Messages on initialization steps: the message 'IPort Not Connected' tells us that 'Site_GUS' is not available yet.
   - '- BCDSAgent:' prompt of Agent's 'Command Line Interface' appears at the last line, or it is buried in previous messages. In either case, type 'Return' on the keyboard.
   - We can see the CLI command listed:
     '*' for Drawing BData
     '>*' for Drawing IPort
     '*>' for Drawing OPort
     '>>*' for Reconnecting IPort
     '>$' for Transferring BData to Application
     '$>' for Updating BData from Application

Step 3: Observing BData, IPort, and OPort of 'Site_uBio'

4. Type '*' (only an asterisk) and 'Return' and see the 'uBio' View (BData of this Site).
   - Only the column names are shown. Left 2 columns for imported data from IPort 'GUS', and Right 2 columns for data to be exported through OPort.
   - Note that the columns for imported data are prefixed by '>' as '>G_n' and '>G_c' which are 'ReadOnly', while exported columns are postfixed by '>!' as 'U_n>!' and 'U_c>!' which may be changed ('ReadWrite') by other Agent.
5. For CLI prompt, type '>*' and 'Return' to see IPort table 'GUS'. This is empty, too at this time.
6. For CLI prompt, type '*>' and 'Return' to see OPort. We can see no tables, simply showing the prompt for next command. OPorts are created in respond to requests from other Agents.

Step 4: Starting 'Site_GUS'

7. Open another new 'Terminal' by clicking 'New Window' and move to 'BCDSAgent/Orchestra' folder, too. And do '../BCDSAgent-exe GUS.conf' for starting 'Site_GUS'.
8. For CLI prompt, type '*' and 'Return' to see BData 'GUS'. We can see the view table which was copied at startup from the table 'GUS' of the database 'CDSS.db'.
   - Note that the column names with '>'-postfix, that is, 'G_c>' and 'G_n>' are assumed to be exported through OPort in responce to the request from other Agents, while column 'G_i' without any postfix is not exported.
   - The column names of corresponding database table 'GUS' are names without any postfixes (and prefixes), i.e., 'G_i', 'G_c', and 'G_n'.
9. Also ask CLI for showing IPort and OPort status by '>*' and '*>'. Both are responded no tables as expected.
   - 'Site_GUS' has no IPort and has not been asked yet to create OPort from other Sites.

Step 5: Connecting IPort of 'Site_uBio' with 'Site_GUS'

10. Move to 'Site_uBio Terminal', and type '>*' again to confirm that IPort 'GUS' has not been connected yet.
11. For CLI prompt, type '>>*' for 'Reconnect IPort'.
    - This operation is necessary also for the case when IPort was closed due to some connection error between Agents. Partner Agent closes its OPort or stops itself, for example.
    - We can recognize the status of the partner Agent only by 'trying to connect to' operation by '>>*'.
12. Type '>*' again to confirm that IPort 'GUS' has been connected to 'Site_GUS' and filled the table 'GUS' of IPort.
13. Also by typing '*' for CLI prompt, we see that IPort 'GUS' contents have been transferred to BData of 'uBio'.
    - Note that the database table 'uBio' is unchanged. The effect of connection is limited to the BCDS View only. We will see how to do with the database.
14. Return to 'Site_GUS Terminal' and observe its OPort by typing '*>'. This shows the same contents of IPort 'GUS' of 'Site_uBio'.

Step 6: Updating BData of 'Site_uBio'

15. Move to 'Site_uBio Terminal', and type '>$' to transfer the BData to Application, i.e., SQLite3.
    - Before this operation, we are advised to confirm that the database 'CDSS.db' is not locked. Press 'Write Changes' button of 'DB Browser for SQLite' saves from uncertain situation.
    - We may use the command line version of SQLite3 starting by '$ sqlite3' from the 'Terminal'. However, the browser is more convenient to use.
16. Use 'DB Browser for SQLite' for updating the BCDS View. We do here make 'ReadOnly' imported data in BData of 'Site_uBio' into its own 'ReadWrite' data to be exchanged with other Agent through OPort.
    - We can see now the Table 'uBio' in the Browser's 'Browse Data' window which displays the same data as BCDS BData View.
    - A small but important difference is observed between database Table 'uBio' and BData View in that the leftmost indexes (record number) are different. This is because indexes of BData View are used for maintaining the View properly in backward transformation to IPorts.

Step 7: Updating Database Table and BData View

17. To update Table 'uBio' of the database, press 'Execute SQL' button of the Browser.
    - We can see new menu bar for SQL processing. Press 'Open SQL File' icon (2nd from the left) to choose the file 'uBio.SQL'.
    - After getting the file, the Browser shows the SQL statement in the window below the icon bar. The SQL code in very short, one line in this case.
    - Pressing 'Rightward Triangle' icon performs the action that SQL states.
18. We can see the result of SQL processing by browsing in 'Browse Data' window. Two records are added.
19. Do not forget to press 'Write Changes' button before leaving the Browser. After writing the changes, the button is grayed out.
20. Return to 'Site_uBio Terminal' and type '$>' for CLI prompt to reflect the update by Application to BData.
21. Type '*' to see the update in BCDS View. This change is not related to imported columns, and hence no propagation occurs to IPort. This is what the 'ReadOnly' columns means. Note that no OPort has created yet, and no propagation occurs, neither.

Step 8: Starting 'Site_BioSQL'

22. Open new 'Terminal' by clicking 'New Window' and move to 'BCDSAgent/Orchestra' folder, too. And do '../BCDSAgent-exe BioSQL.conf', starting 'Site_BioSQL. This procedure is same as one on starting 'Site_GUS' in Step 4.
23. For CLI prompt, type '*' and 'Return' to see BData 'BioSQL'. This time, we can see the view with 2 records.
    - Contents of BData 'BioSQL' come from IPort 'uBio' of this Agent. Type '>*' for CLI to see the IPort which has been connected to an OPort of 'Site_uBio'.
    - OPort of 'Site_uBio' has been created in respond to request from 'Site_BioSQL'.
24. To confirm the OPort of 'Site_uBio', move to 'uBio Terminal' and type '*>'. We can see OPort View containing the same contents as IPort 'uBio' of 'Site_BioSQL'.

Step 9: Updating BData of 'Site_BioSQL' by Application

25. On 'Site_BioSQL Terminal', type '>$' to transfer the BData to SQLite3.
    This Step is similar to Steps 6 and 7 for 'Site_uBio'.
    - Before this, confirm that the database 'CDSS.db' is not locked.
    - We can see now Table 'BioSQL' in the Browser's 'Browse Data' window which displays the same data as BCDS BData View.
26. To update the table 'BioSQL', press 'Execute SQL' button of the Browser and get 'BioSQL.SQL' file. The code shown in the window is SQL statements of several lines.
    - Pressing 'Rightward Triangle' icon as before to process SQL.
    - We move to Table 'BioSQL' in 'Browse Data' window and see that a new record is added at the bottom.
    - Do not forget to press 'Write Changes' button before leaving the Browser.
27. Return to 'Site_BioSQL' and type '$>' to get the updated view for BData.

Step 10: Observing Bidirectional Update

28. On 'Site_BioSQL Terminal', type '*' to see BData of 'Site_BioSQL'. We see that a new record appears there.
29. Next move to 'Site_uBio Terminal' and type '*' to see BData of 'Site_uBio', where we can see a new record added at the bottom with Right 2 columns having non-'None' values. This is the result of synchronization, or propagation of the View-update by 'Site_BioSQL'.
30. To observe the situation, confirm IPort of 'Site_BioSQL' and OPort of 'Site_uBio'. And review the Log messages with time to see what makes Bidirectional View-update.
31. To end our exercise, press 'Control+C' on 'GUS Terminal'. This stops 'Site_GUS', and hence close its OPort. This causes closed the connection with IPort of 'Site_uBio'.
    - We can see the message of this event on 'Site_uBio'. We can reconnect as in Step 5 after 'Site_GUS' is restarted.
32. To stop Site, press 'Control+C'. No other key is provided for termination.

Final Remark: We have been working on a single computer for experiments. However, we can deploy Agents over different machines on the LAN by specifying IP-address in the configuration file. Try!

Another Exercise: Dejima Configuration

```
-- DejimaXYZ
#BData DejimaXYZ DejimaDemo.db 55555 >X >Y >Z
#IPort DejimaX localhost 55556 >X
#IPort DejimaY localhost 55557 >Y
#IPort DejimaZ localhost 55558 >Z
```

All Columns are impo

IPort 'DejimaY' to be connected to <localhost::55557>

See diagrams on next page

```
-- DejimaSite_X
#BData Site_X DejimaDemo.db 55556 X> >Y >Z
#IPort DejimaYZ localhost 55555 >Y >Z
```

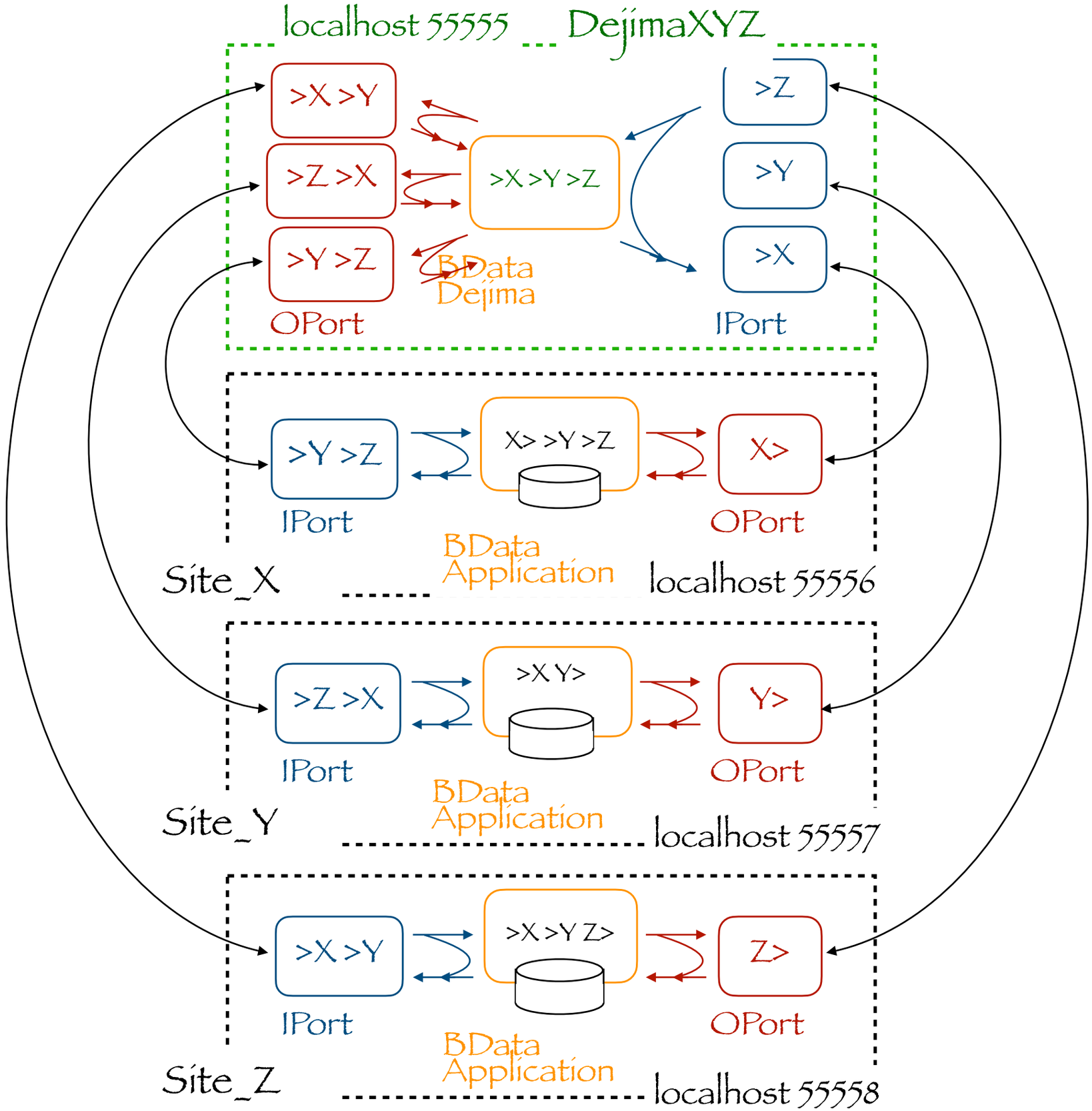
BData consists of Own Data 'X' and imported 'Y' and 'Z' in ReadOnly mode.

```
-- DejimaSite_Y
#BData Site_Y DejimaDemo.db 55557 >X Y> >Z
#IPort DejimaZX localhost 55555 >Z >X
```

```
-- DejimaSite_Z
#BData Site_Z DejimaDemo.db 55558 >X >Y Z>
#IPort DejimaXY localhost 55555 >X >Y
```

IPort 'DejimaXY' to be connected to <localhost::55555>

Configuration of BCDS Sites with Dejima



Exercise on Running BCDS of Dejima Style

Steps

1. Starting Dejima Agent and Application Sites
    - Starting Agents ①Site_X, ②DejimaXYZ, ③Site_Y, and ④Site_Z in this order.
    - Note that the necessary Ports for Agents to connect may be unavailable at starting time. For example, Site_X tries to connect IPort 'DejimaYZ' of DejimaXYZ Agent, but fails because it has not been started at that time.
2. Refresh/Reconnect Ports each other
    - After all the necessary Agents have been started, reconnect IPorts which have not connected yet. This process is inevitable for configuration with mutual references of data Sites.
3. Updating Data View by Application
    - View updating on Application Site_Z using Database Browser: in-place update of a cell value, deletion of a record, and addition of a new record.
    - Updating results are propagated to related Ports and all the participant Sites without any intervention

Reference: Configuration Specification of BCDS Agent

```
--BCDS Site_A Specification {Note on Agent; no effect on configuration}
#BData <BDataName> <DatabaseName> <OPortNumber> <Columns>
#IPort <IPortName> <HostAddr> <PortNumber> <Columns>
... {#IPort specification may be repeated}
```

<BDataName> : Agent name corresponding to 'Table Name' of the Database.

<DatabaseName> : Name of Local Database which maintains Agent's Own data.

<OPortNumber> : PortNumber with which Agent is listening on for connection requests.

<IPortName> : Name of IPort

<HostAddr>, <PortNumber>: Host address and PortNumber of other Agent; Agent will try to request connection for this IPort.

<Columns> : A sequence of <ColumnName>

<ColumnName> : Name of column name of BData Table, which consists of <Prefix>, 'DbColName', and <Postfix>. 'DbColName' corresponds to the column name of the local database.

<Prefix> : Either ">" or ">!"; ">" for the column of ReadOnly imported data, and ">!" for ReadWrite ones. These prefix are used for columns of IPorts. Data of columns prefixed with ">" cannot be updated by BData Application, while ones prefixed with ">!" can be updated.

<Postfix> : Either ">" or ">!"; exports Agent's Own data through OPort with ReadOnly mode (">") or ReadWrite mode (">!"). These postfixes are matched when creating OPorts for connection requests from other Agent's IPort columns with prefixed <ColumnName>.