Tivoli software

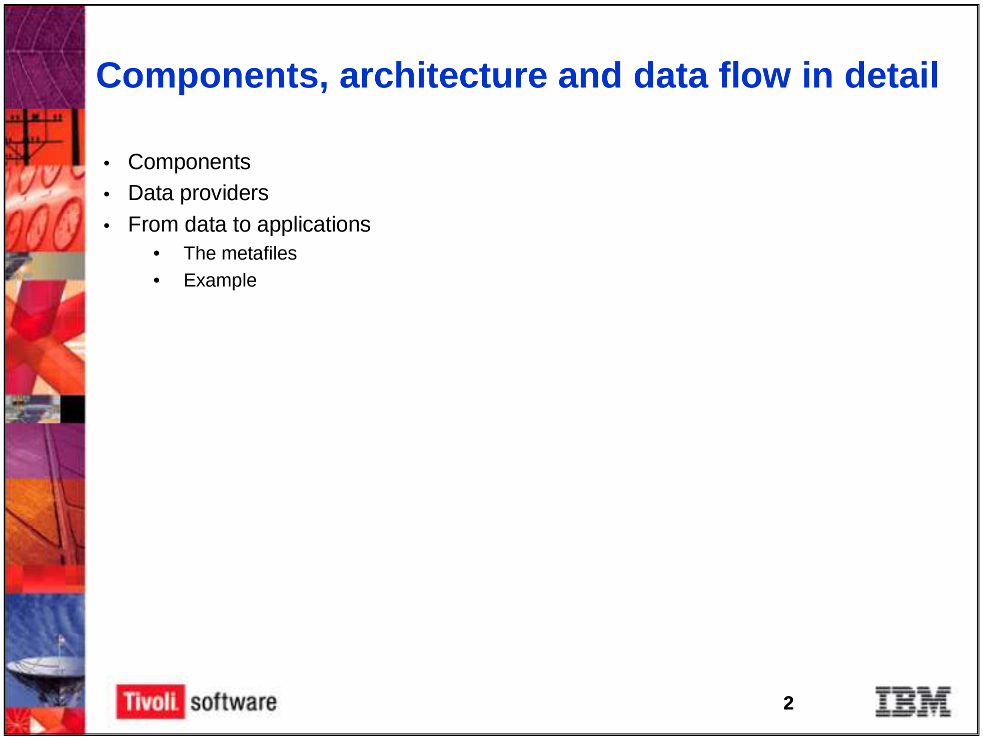
IBM

IBM Tivoli Monitoring 6.1 Universal Agent
Components, architecture and data flow in detail

e business software

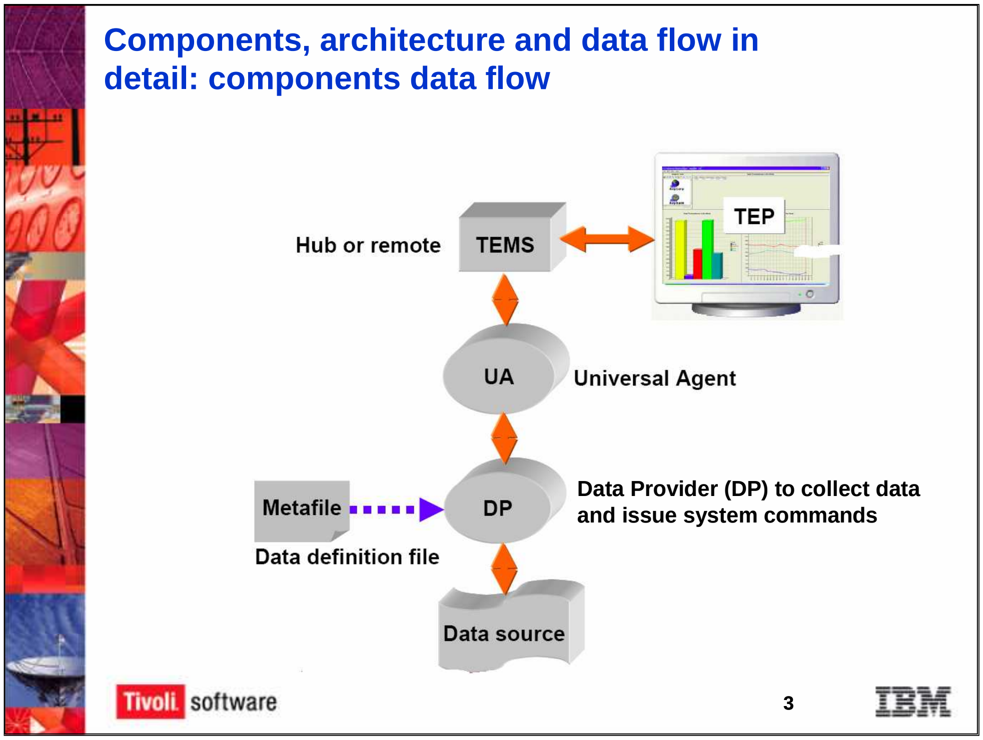
IBM Software Group

IBM Tivoli Monitoring 6.1 Universal Agent
Components, architecture and data flow in detail



Components, architecture and data flow in detail

- Components
- Data providers
- From data to applications
  - The metafiles
  - Example



Components, architecture and data flow in detail: components data flow

Hub or remote TEMS

TEP

UA Universal Agent

Metafile

Data definition file

DP

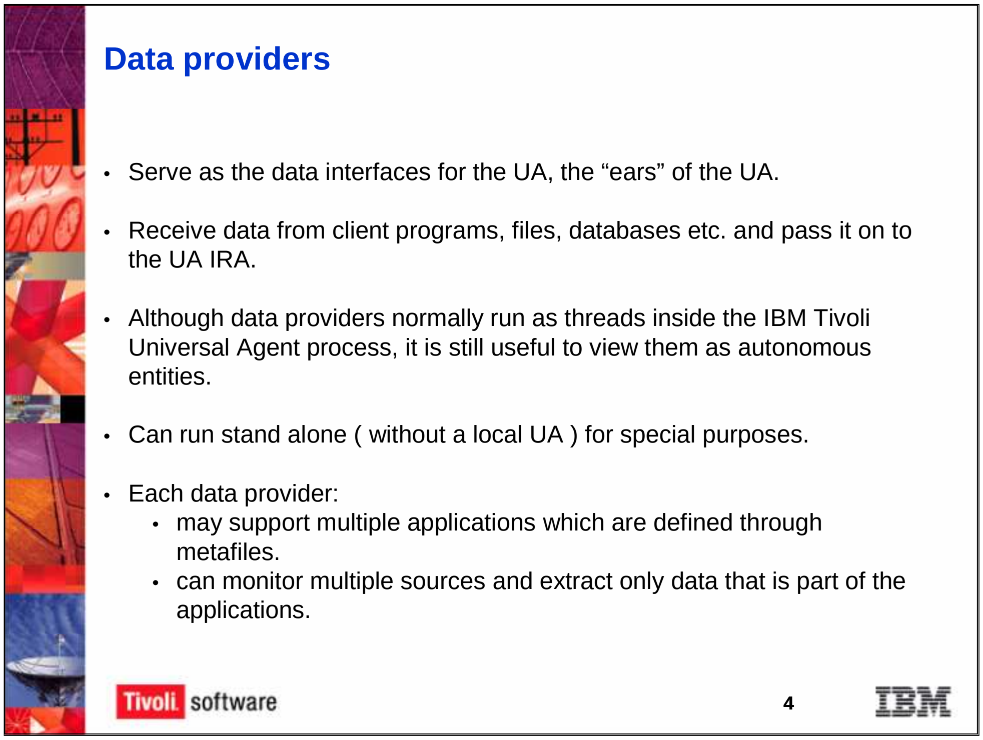
Data Provider (DP) to collect data and issue system commands

Data source

Tivoli software

3

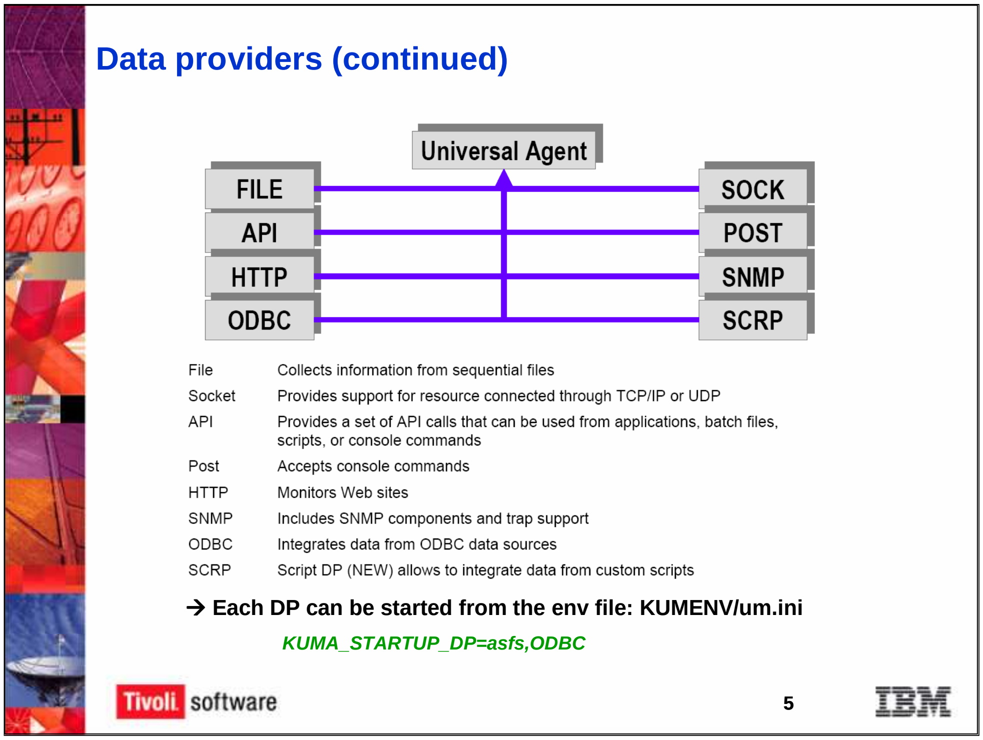
We'll go over the components, architecture and data flow in some detail. It's got your TEMS hub or remote. The agent connects to the TEMS and on startup the agent will start the data providers that it has requested to start. If it has no data providers in the startup line, then it'll pretty much be a running process with no data providers running. The SNMP emitter will run. Other than that, it will not load any applications. But the data providers are started in my default. It starts the AFSF, which is basically the file provider and the script provider. When they load, they then look in the config. file of the agent to see which metafiles to load. A metafile is a data definition file. Those are files that you write or somebody writes for you and then that information is then loaded by the data provider. Then it begins to communicate with this data source, whatever that data source may be.



Data providers

- Serve as the data interfaces for the UA, the “ears” of the UA.
- Receive data from client programs, files, databases etc. and pass it on to the UA IRA.
- Although data providers normally run as threads inside the IBM Tivoli Universal Agent process, it is still useful to view them as autonomous entities.
- Can run stand alone (without a local UA) for special purposes.
- Each data provider:
  - may support multiple applications which are defined through metafiles.
  - can monitor multiple sources and extract only data that is part of the applications.

The data providers, they service the data interfaces for the universal agent. They do the collection. They do the interaction with the product that you are looking to monitor. They receive data from client programs. They receive data from files, databases, etc. and pass that on to the universal agent. They do run within the agent. They are not separate threads, they’re not separate processes. Although the data… it says right there. They can run stand alone, without a local UA, for special purposes. Each Data provider supports multiple applications as defined through the metafiles. They can monitor multiple sources and extract only data that is part of the application.



Data providers (continued)

Universal Agent

FILE

API

HTTP

ODBC

SOCK

POST

SNMP

SCRP

File	Collects information from sequential files
Socket	Provides support for resource connected through TCP/IP or UDP
API	Provides a set of API calls that can be used from applications, batch files, scripts, or console commands
Post	Accepts console commands
HTTP	Monitors Web sites
SNMP	Includes SNMP components and trap support
ODBC	Integrates data from ODBC data sources
SCRP	Script DP (NEW) allows to integrate data from custom scripts

→ Each DP can be started from the env file: KUMENV/um.ini

KUMA_STARTUP_DP=asfs,ODBC

Tivoli software

API

The Application Programming Interface Server (APIS) Data Provider supports API client functions. The data provider and the IBM Tivoli Universal Agent APIs enable you to easily develop scripts and C/C++ programs that send data to the IBM Tivoli Universal Agent. They also support a command-line interface which implements a subset of the API functions.

→ Asfs Consolidates API, Socket, File, Script (ASFS)

Post:

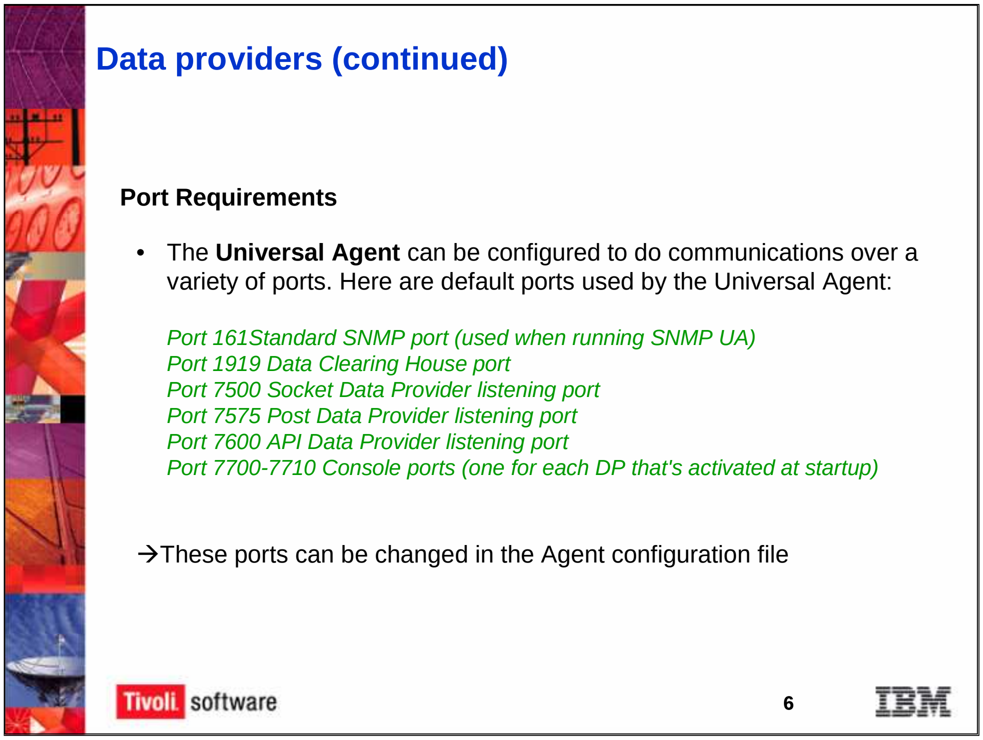
The Post Data Provider offers a convenient means of sending ad hoc notifications such as messages, alerts, and status information to IBM Tivoli Monitoring.

Socket:

The Socket Data Provider enables the IBM Tivoli Universal Agent to manage data on operating systems where it cannot be installed by implementing program-to-program communication using the socket transport paradigm. Figure 11 illustrates the role of Socket Data Provider. **(This is from the UG)**

These are the data providers that are available. There is the File, Socket, API, Post, HTTP, SNMP, ODBC. SCRP. It is important to note that in the ini file, or in the Windows environment its KUMENV file, there is a parameter, KUMA_STARTUP_DP.

That tells the agent what data providers to start. Now on the UNIX operating system and Linux operating system, you can actually provide it a parameter on startup. You can give it a minus-0 option and tell it which ones that you want to start, and it will start those. The difference between setting the variable and starting it with minus-0 is that if you don't use the minus-0 identical every time, you won't start the same providers every time. If you use the variable, then those are the ones that are going to start. Make sure on UNIX that you modify the um.ini not the um.config file. That way it will maintain that for your next start.



Data providers (continued)

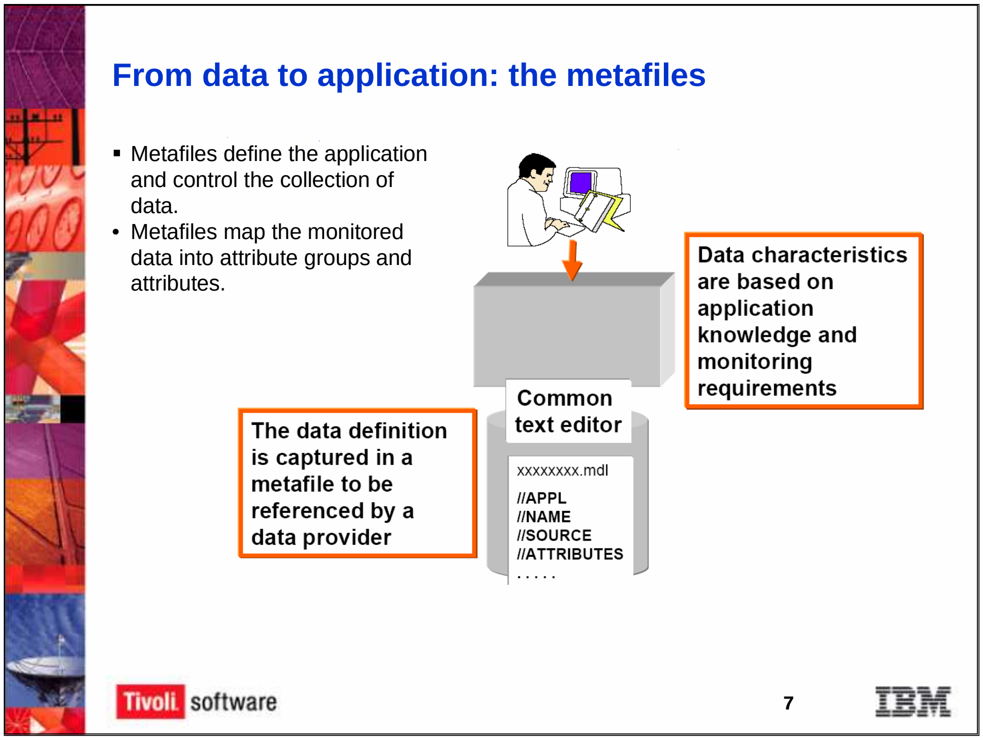
Port Requirements

- The **Universal Agent** can be configured to do communications over a variety of ports. Here are default ports used by the Universal Agent:

  Port 161Standard SNMP port (used when running SNMP UA)
  Port 1919 Data Clearing House port
  Port 7500 Socket Data Provider listening port
  Port 7575 Post Data Provider listening port
  Port 7600 API Data Provider listening port
  Port 7700-7710 Console ports (one for each DP that's activated at startup)

→These ports can be changed in the Agent configuration file

Here are some important considerations. These are the default ports that the agent uses to talk. It does use port 161 for SNMP. If you have SNMP running on the machine already using that port then you need to switch the SNMP ******* universal agent or turn off SNMP to another port. The data clearinghouse port, if this one does not get 1919 unless you change it to another port the agent will not start. It will shut itself down. Your other ports; the socket, post, API, console. The agent can actually start with those ports not being available. Now those providers, if they're listening on those ports and the ports aren't available, then they're not going to work. So those are notes. And those do get logged. It will say that he brought up the agent on port such-and-such and it will say that it got connected or it will say that it didn't. Say 'port 161 is already in use. SNMP data provider is not starting'. That kind of information is in the log, so that way you know why you're not getting data on certain ports. But you do need to make sure these ports are available or you need to switch and use the environment variables out of the users guide.



From data to application: the metafiles

- Metafiles define the application and control the collection of data.
- Metafiles map the monitored data into attribute groups and attributes.

Data characteristics are based on application knowledge and monitoring requirements

Common text editor

The data definition is captured in a metafile to be referenced by a data provider

```
xxxxxxxx.mdl
//APPL
//NAME
//SOURCE
//ATTRIBUTES
.....
```

Tivoli software 7 IBM

Everything UA knows about an application comes from the metafile.

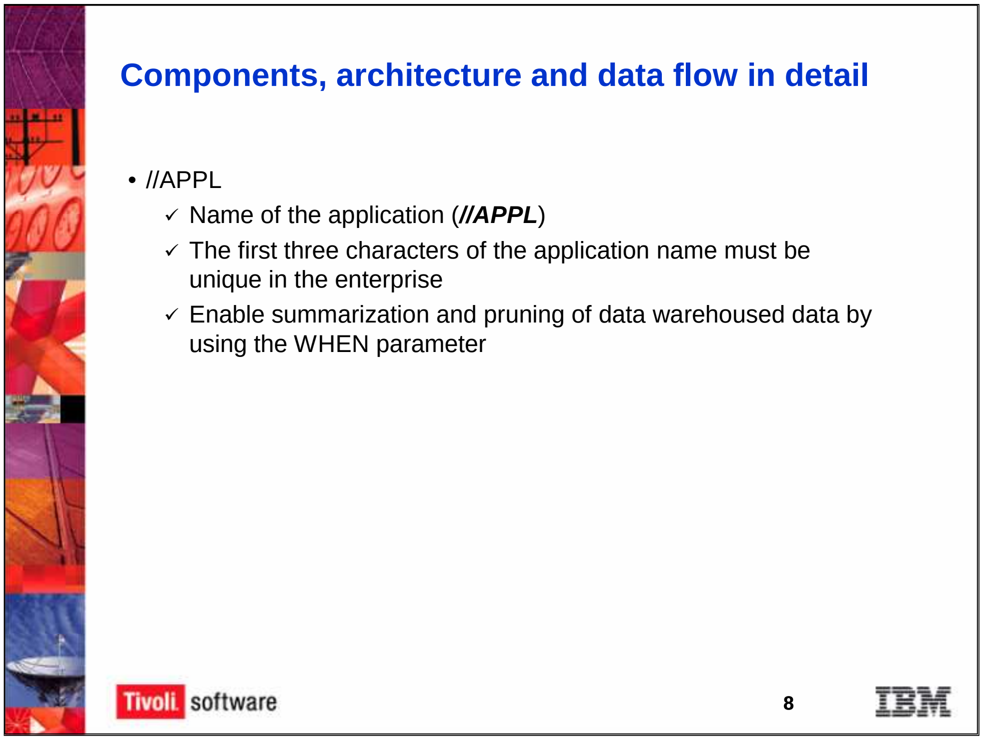
Metafiles map the layout of input data, splitting the data into attributes that can be viewed or referenced in situations and policies.

Under //NAME and //SOURCE, you define the individual attributes.

You can have multiple metafiles active in a single UA system.

KUMPCNFG stores active metafiles

The data is written into a metafile. Metafile defines the application and controls a collection of data. It maps the monitor data and attribute groups and attributes, so you get to define your attributes and attribute groups as you define your metafiles. So the data characteristics are based on your application knowledge, your monitoring requirements. You need to look through the users guide and pay attention to the different attribute types and the different data they provide based on what kind of data you need to view in the TEPS or also write reports for. The user’s guide will be your friend as you work with the universal agent.



Components, architecture and data flow in detail

- //APPL
  - ✓ Name of the application (***//APPL***)
  - ✓ The first three characters of the application name must be unique in the enterprise
  - ✓ Enable summarization and pruning of data warehoused data by using the WHEN parameter

Tivoli software 8

A metafile can contain only one application definition statement (//APPL xxxxx). **The first 3 characters of the application name must be unique in the enterprise**.

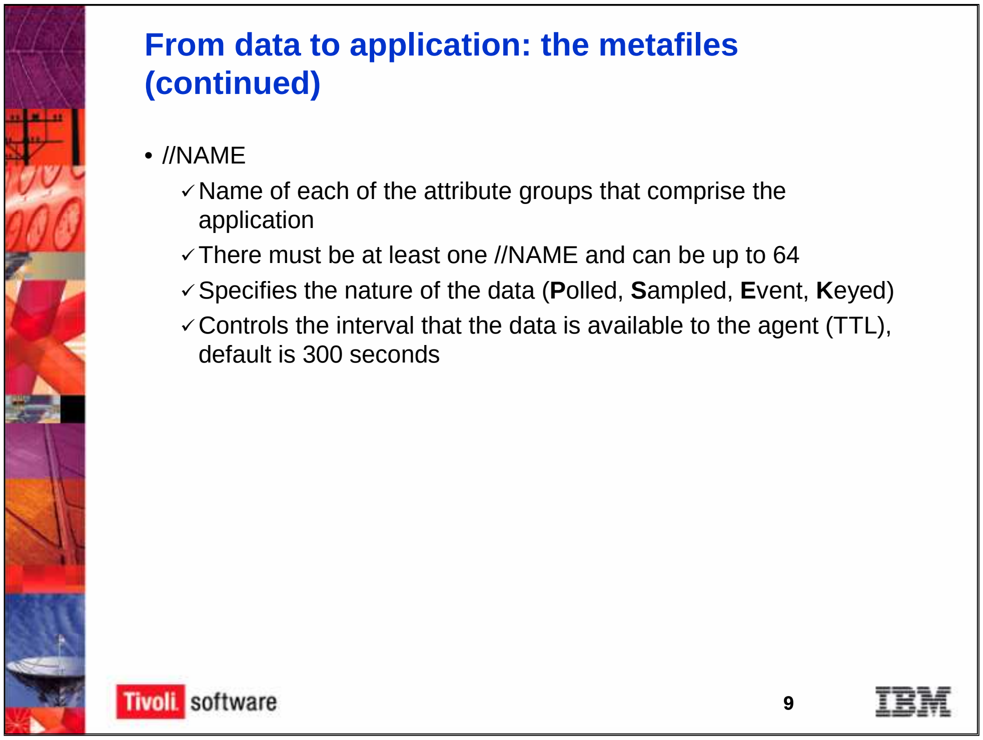
A metafile can have multiple attribute groups which are synonymous with application tables (//NAME statements).

Attribute groups can have many data sources (//SOURCE statements).

//NAME statement specifies application data type: polled, sampled, event, or keyed, and time-to-live (TTL) value of collected data rows.

Customers can define their own attribute help text in metafiles.

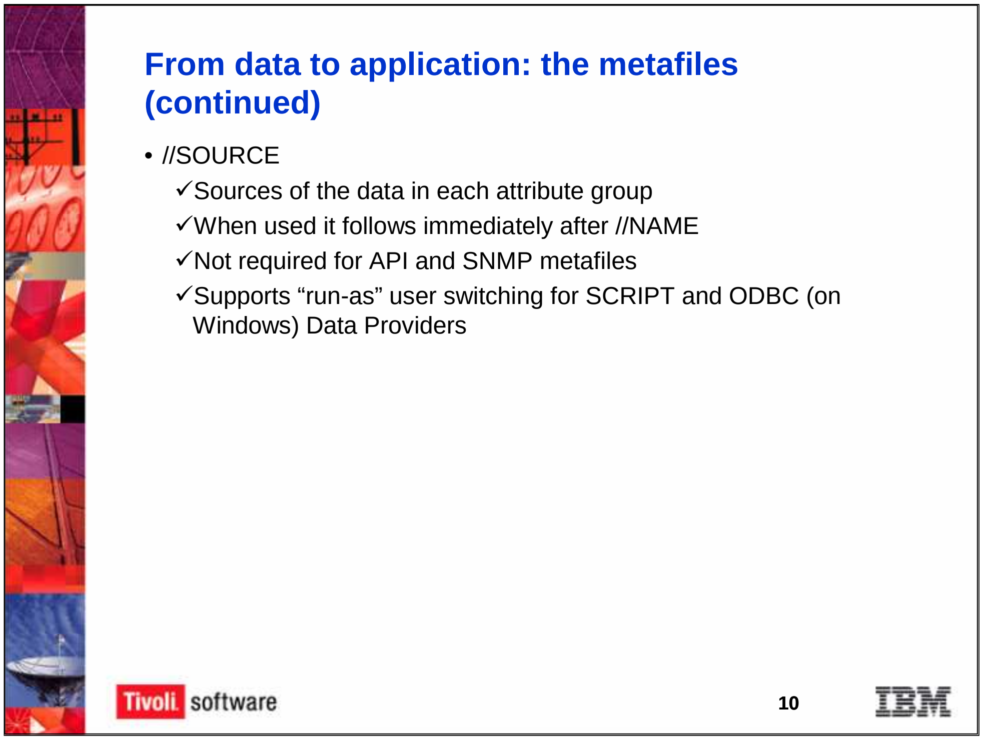
We're going to go over some of the keywords that are required in a metafile. The only metafile that will be different is the SNMP metafile and cannot have this //APPL in it. This names the application. It's the first three characters and it must be unique within the enterprise, if it's not, then it won't use it. If you want to summarize and prune data then you need to use the WHEN parameter and if it's not in there then data can be historically collected but you can't summarize and prune it within the warehouse.



From data to application: the metafiles (continued)

- //NAME
  - ✓ Name of each of the attribute groups that comprise the application
  - ✓ There must be at least one //NAME and can be up to 64
  - ✓ Specifies the nature of the data (**P**olled, **S**ampled, **E**vent, **K**eyed)
  - ✓ Controls the interval that the data is available to the agent (TTL), default is 300 seconds

The name parameter controls the attribute groups. There must be at least one name and it can be up to 64 per metafile. This specifies the nature of the data (Polled, Sampled, Event, Keyed). Again, I can’t stress enough the use and understanding of the user’s guide because these all have details in there that I don’t have at the presentation since I didn’t see the need to rewrite the users guide into this presentation. The control interval is also in here, the default is 300 seconds. If it doesn’t have a TTL in there, then it’s going to default to that. If you want to change the interval then you would change it in here.



From data to application: the metafiles (continued)

- //SOURCE
  - ✓Sources of the data in each attribute group
  - ✓When used it follows immediately after //NAME
  - ✓Not required for API and SNMP metafiles
  - ✓Supports “run-as” user switching for SCRIPT and ODBC (on Windows) Data Providers

Tivoli software

10

PSWD is for Windows only

Agent must run as root on UNIX to support USER

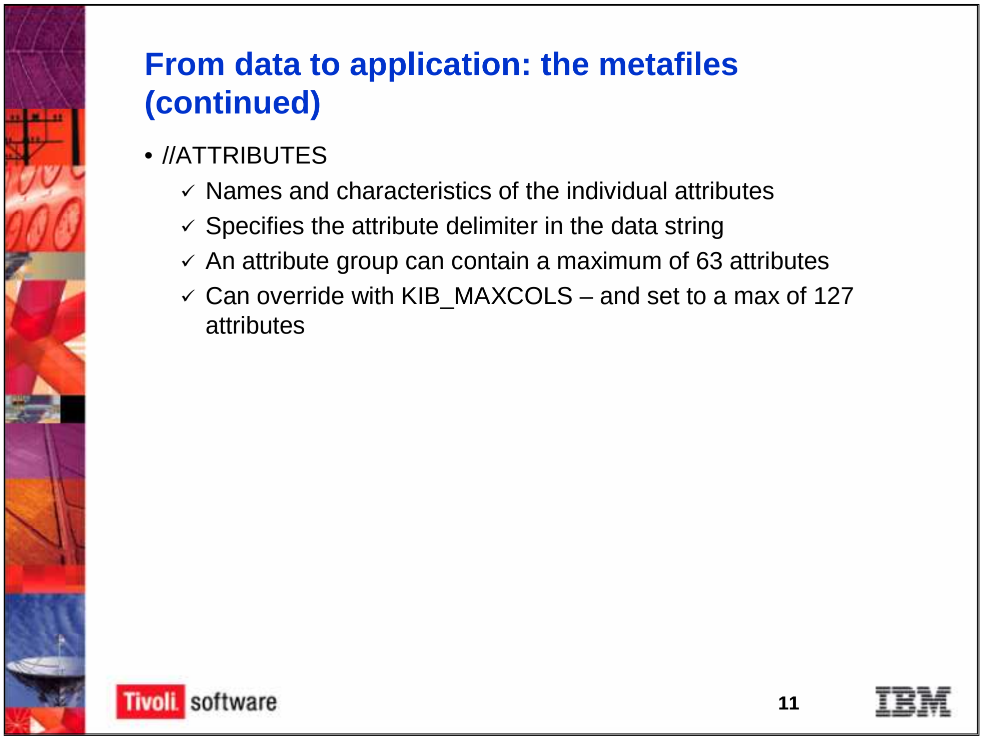
On UNIX/Linux, scripts run with a default PATH of

usr/bin:/bin:/usr/local/bin:/usr/sbin

Because many scripts need environment variables set in order to run properly, the Script DP supports an “envfile=xxxx” parameter on the //SOURCE SCRIPT statement. This envfile must contain a series of variable=value statements, one per line.

Arguments are placed in double quotes

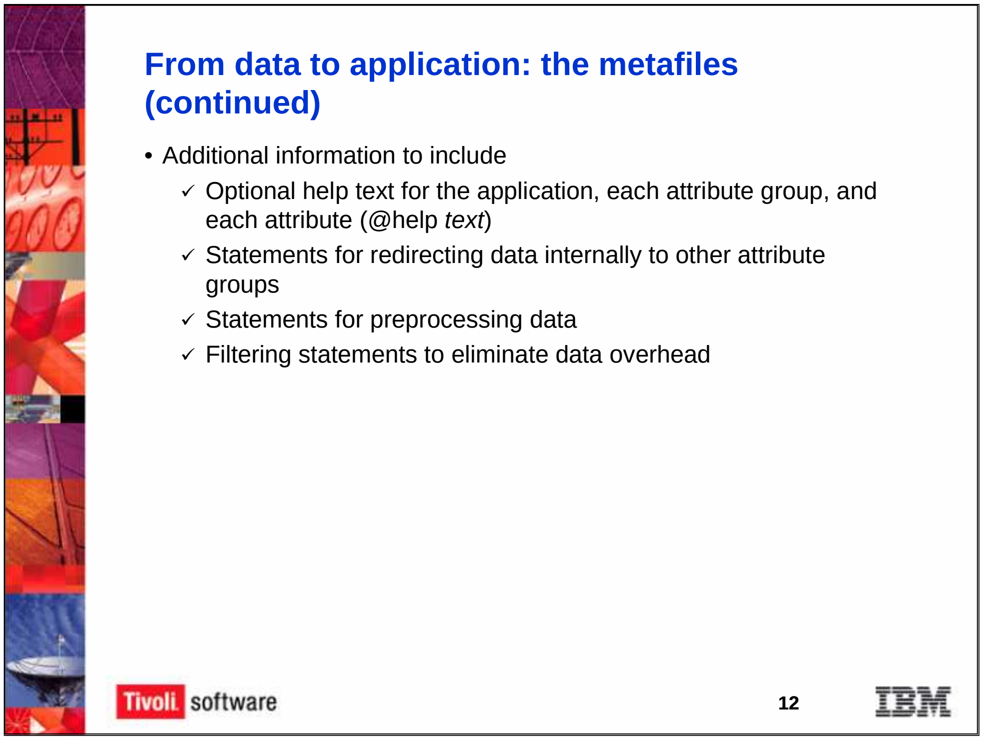
This is the one that I mentioned earlier for //APPL was the SNMP. This is the one that is not required for SNMP. The source tells it what kind of data this is going to be, where is it going to come from? The source for every attribute group and it immediately follows the //name, so you can use it with script and ODBC. On Windows you can actually tell it to run as another user. The agent itself has to run with enough authority to switch the user context. On Unix it would have to run as root, on Windows it would have to run as an administrator, otherwise it won’t be able to switch context when that’s used. Do pay attention to the user’s guide here, because it is… when you do as and have it run as a user you do need to be space sensitive as to how it’s set up. The syntax needs to be followed; otherwise it won’t adhere to that.



From data to application: the metafiles (continued)

- //ATTRIBUTES
  - ✓ Names and characteristics of the individual attributes
  - ✓ Specifies the attribute delimiter in the data string
  - ✓ An attribute group can contain a maximum of 63 attributes
  - ✓ Can override with KIB_MAXCOLS – and set to a max of 127 attributes

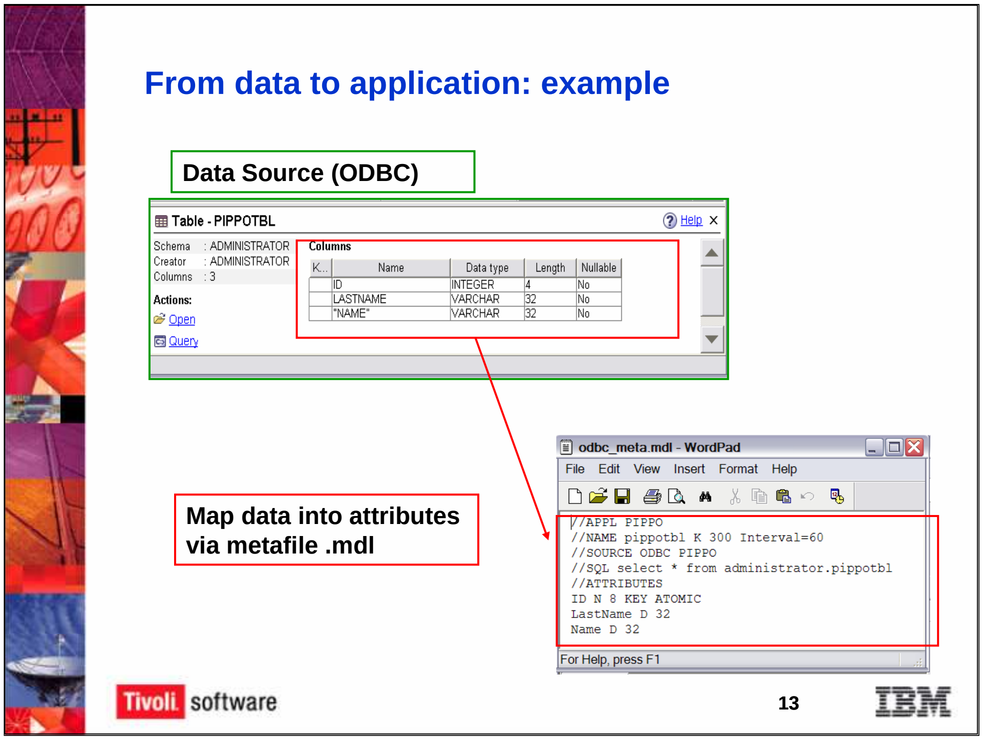
We’re going to go on to Attributes. This actually names and characteristics of the individual attributes that you’ll see up in the TEP or in the warehouse, depending on where you’re going to view them. This specifies the delimiter of the data string. An attribute group itself contains 63 attributes by default. You can override that by setting this variable, KIB_MAXCOLS – and that can be set up to 127. That’s if you need more that 63 attributes within an attribute group. In fact, I believe that in UNIX that may be the default, is to set it at that, but that is something you can check in the .ini file. If it is set then it will override the default of 63.



From data to application: the metafiles (continued)

- Additional information to include
  - ✓ Optional help text for the application, each attribute group, and each attribute (@help *text*)
  - ✓ Statements for redirecting data internally to other attribute groups
  - ✓ Statements for preprocessing data
  - ✓ Filtering statements to eliminate data overhead

Some additional information to include. You can actually have optional help text by putting an ‘@help’ at the end of a line and in text and that will be available in the ODI file and in the CAT and ATR files makes it available to them. You can redirect data internally to other attribute groups. You can do preprocessing of the data, so you can filter it before it actually hits your UA and gets up to the TEP. And then this can eliminate some of your data overhead so that the agent isn’t seeing data that you don’t want it to see.



From data to application: example

Data Source (ODBC)

Schema : ADMINISTRATOR
Creator : ADMINISTRATOR
Columns : 3

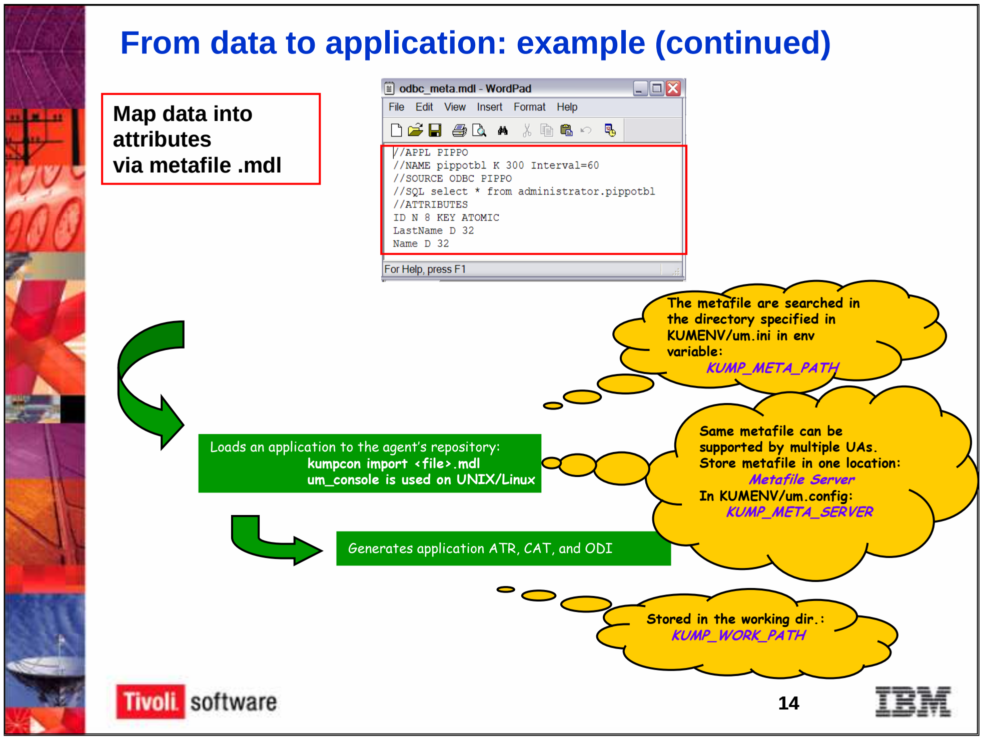
Columns

K...	Name	Data type	Length	Nullable
	ID	INTEGER	4	No
	LASTNAME	VARCHAR	32	No
	"NAME"	VARCHAR	32	No

Map data into attributes via metafile .mdl

```
//APPL PIPPO
//NAME pippotbl K 300 Interval=60
//SOURCE ODBC PIPPO
//SQL select * from administrator.pippotbl
//ATTRIBUTES
ID N 8 KEY ATOMIC
LastName D 32
Name D 32
```

In this sample we're going to look at an ODBC data source. It's a very simple table. You can see the definition here. It's got name, data type, link, ID, and last name, and name. These map into the attributes to be a metafile, which is an .mdl. So we're going to call it //pippo and we're going to go off of this table in an interval of 300 seconds, we're going to collect this data every 60 seconds. And the source is going to be an ODBC data source called pippo. Now, one thing to note with this is that you need to write this data source yourself. The UA will not generate a data source on a Microsoft machine, you need to go in and create that yourself and verify that it will connect to the table that you want to use. UA can automatically generate metafiles using the ODBCA source with a setting but it will not automatically generate the data source itself. Then, if this table had any restrictions like user ID and password to connect to this database, that would be on the source line as well. Currently you would just say user equals the username and password equals the password. Then you would put in your SQL statement and it will select from this table administrator.pippotable. In the attributes, it's going to be ID, the numeric 8, last name and a display numeric of 32 and name of a display of 32.



From data to application: example (continued)

Map data into attributes via metafile .mdl

```
//APPL PIPPO
//NAME pippotbl K 300 Interval=60
//SOURCE ODBC PIPPO
//SQL select * from administrator.pippotbl
//ATTRIBUTES
ID N 8 KEY ATOMIC
LastName D 32
Name D 32
```

The metafile are searched in the directory specified in KUMENV/um.ini in env variable: *KUMP_META_PATH*

Loads an application to the agent's repository: **kumpcon import <file>.mdl um_console is used on UNIX/Linux**

Same metafile can be supported by multiple UAs. Store metafile in one location: *Metafile Server* In KUMENV/um.config: *KUMP_META_SERVER*

Generates application ATR, CAT, and ODI

Stored in the working dir.: *KUMP_WORK_PATH*

Page 23 of UA UG for KUMP_META_SERVER

The UA would actually connect up to this table and map these attribute files across. Now, if I'm doing this, you can do an import. I usually do a verify first to make sure that my syntax is right and then I do an import. The kumpcon command is what you use on Windows and you run um_console on UNIX or Linux. Then it will generate the ATR, CAT, and ODI files and load those up to the TEMS and TEPS. The metafile directory is specified by a variable in the .ini file called KUMP_META_PATH. The default is ITM home TMA ITM 6. Work for… is it work, let me look. I think it's work under Windows and on UNIX it's the ITM home, then the platform, for example AIX513/um and then under there is the metafiles directory and the work directory. You can change it to wherever you'd like, but that's the default. You can also set up a meta-server and it will connect to that server. Obviously the machine has to be able to get to that server in order to download metafiles for it, which you can set that up.

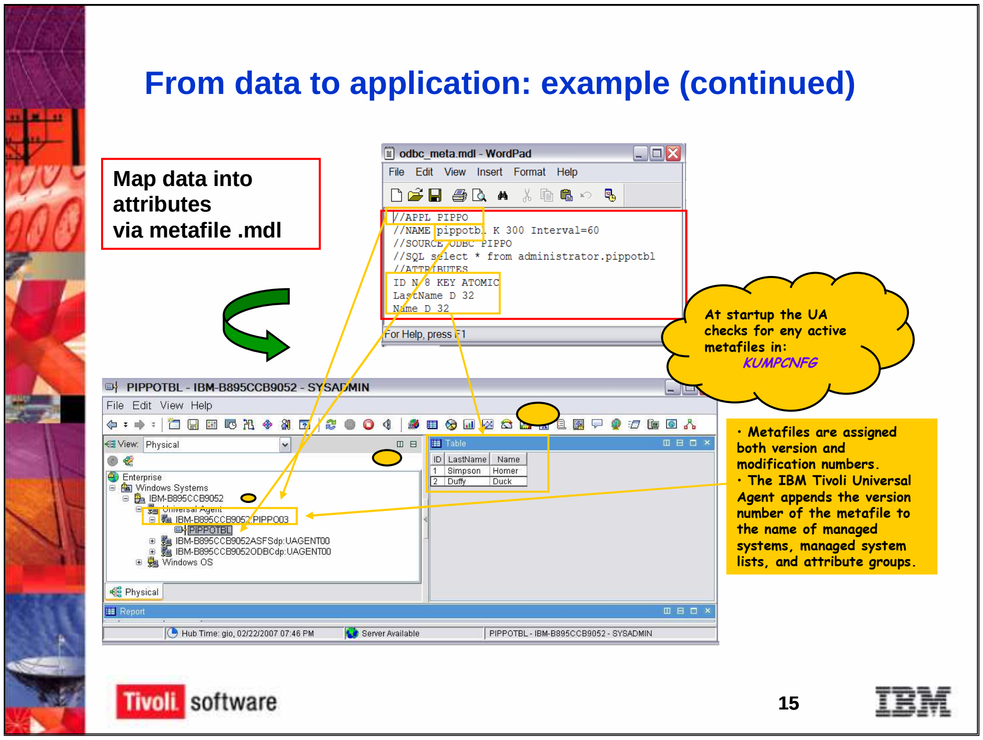
Scott, I think we have a question from Marco. The question is:

When would you need a user name and password from ODBC connection?

Any time a database requires… I mean obviously the one that's shown in here… the database belongs to the administrator and the administrator is the one who's running this agent. But for example, if I write one that connects to the TEPS and I want to collect data out of the TEPS, the TEPS is going to a… the default will say ITM user and the universal agents running as administrator then I would change that source statement. I would say user equals ITM users, space, PSWD equals whatever the password is and right now that password is clear text. It does not get encrypted. There is a request open to change that such that it would get encrypted, but right now you would put that on the source line so it would then connect as that user to the database. If you're connecting as a user that doesn't own the tables but has access to the tables then you need to fully qualify the tables, otherwise the user… you know, if you connect as ITM user, and ITM user owns the tables, you may not have to fully qualify them. If I have it, I would fully qualify mine. That way I don't accidentally pick up any other tables. I hope that helps.

And keep in mind that you can also write it to a remote database as long as the ODB source on the Windows machine can connect to the remote database.

Ok, and the next thing that the slide is showing you is that these CAT and ATR ODI files are actually stored in the work path. The ODI files is for the TEPS and defines how the TEPS will set up the workspace and reports and the CAT and ATR files are for the TEMS, however they do get copied on both machines, or servers.



From data to application: example (continued)

Map data into attributes via metafile .mdl

```
//APPL PIPPO
//NAME pippotbl K 300 Interval=60
//SOURCE ODBC PIPPO
//SQL select * from administrator.pippotbl
//ATTRIBUTES
ID N 8 KEY ATOMIC
LastName D 32
Name D 32
```

PIPPOTBL - IBM-B895CCB9052 - SYSADMIN
File Edit View Help
View: Physical
Enterprise
Windows Systems
IBM-B895CCB9052
Universal Agent
IBM-B895CCB9052:PIPPO03
PIPPOTBL
IBM-B895CCB9052ASFSdp:UAGENT00
IBM-B895CCB9052ODBCdp:UAGENT00
Windows OS
Physical
Table
Report
Hub Time: gio, 02/22/2007 07:46 PM
Server Available
PIPPOTBL - IBM-B895CCB9052 - SYSADMIN

ID	LastName	Name
1	Simpson	Homer
2	Duffy	Duck

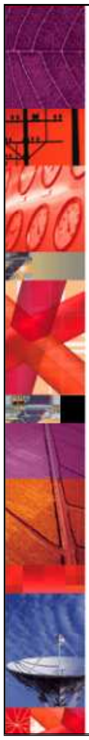
At startup the UA checks for eny active metafiles in: *KUMPCNFG*

- Metafiles are assigned both version and modification numbers.
- The IBM Tivoli Universal Agent appends the version number of the metafile to the name of managed systems, managed system lists, and attribute groups.

Active metafiles stored in KUMPCNFG

At startup, the IBM Tivoli Universal Agent checks to see if any metafiles are specified in the KUMPCNFG configuration file. If so, the IBM Tivoli Universal Agent loads the appropriate metafiles from the path names specified in KUMPCNFG, or if fully qualified names were not specified, from the metafiles directory. You can also use the IMPORT and REFRESH console commands to import new and revised metafiles at any time.

We see that the data that we've written this application for is now collected in and we look in the middle to lower left of the screen and you can see the TEP and you actually can see that it has generated a workspace and it's given it the name that we gave our application. We come under and we'll see that the name of… and it has a version number, and on this one it's version 03 so this has actually been modified or reloaded at least three times to get this in there. The default is zero zero, so it'll be PIPO00. It has the machine name which is the long number you see before that. And then you get the application name and the version. Underneath that you see the attribute groups. It comes down and fills these in. And by default, and it's kind of in order that we could get this graphic on this slide, you see where it says report down there underneath the word physical, in that window. That's going to be the actual information where the actual attributes are. And in this case we've actually gone in and put a table over here to the right. By default, you won't have a table on the right. By default you'll have a report below and I believe the notepad or to the right will be blank. You can change it how you want. It is a workspace, you can pull in data as you see fit. But it will put every attribute in that table or the report below. You can write situations at this point, I don't believe we go over that in here, but you could then… the data that we have in here doesn't truly lend itself to a situation, but if we want to write a situation that says, you know, fire off a critical situation where last name equals Simpson and first name equals Homer. You could open up situation editor by right clicking PIPO table and go into that situation editor and select those attributes and say last name is equal to Simpson and name equals Homer. I need a critical situation. Now, probably in a real world example you would have something different, but what we've got here, it would then fire a situation up at the enterprise that would let you know that Homer Simpson has now been inserted into this table. And that's the kind of situation where you get into being able to use it for real world type of situations and issues.



Copyright and trademark information

IBM trademarks

http://www.ibm.com/legal/copytrade.shtml#ibm

Fair use guidelines for use and reference of IBM trademarks

http://www.ibm.com/legal/copytrade.shtml#fairuse

General rules for proper reference to IBM product names

http://www.ibm.com/legal/copytrade.shtml#general

Special attributions

http://www.ibm.com/legal/copytrade.shtml#section-special