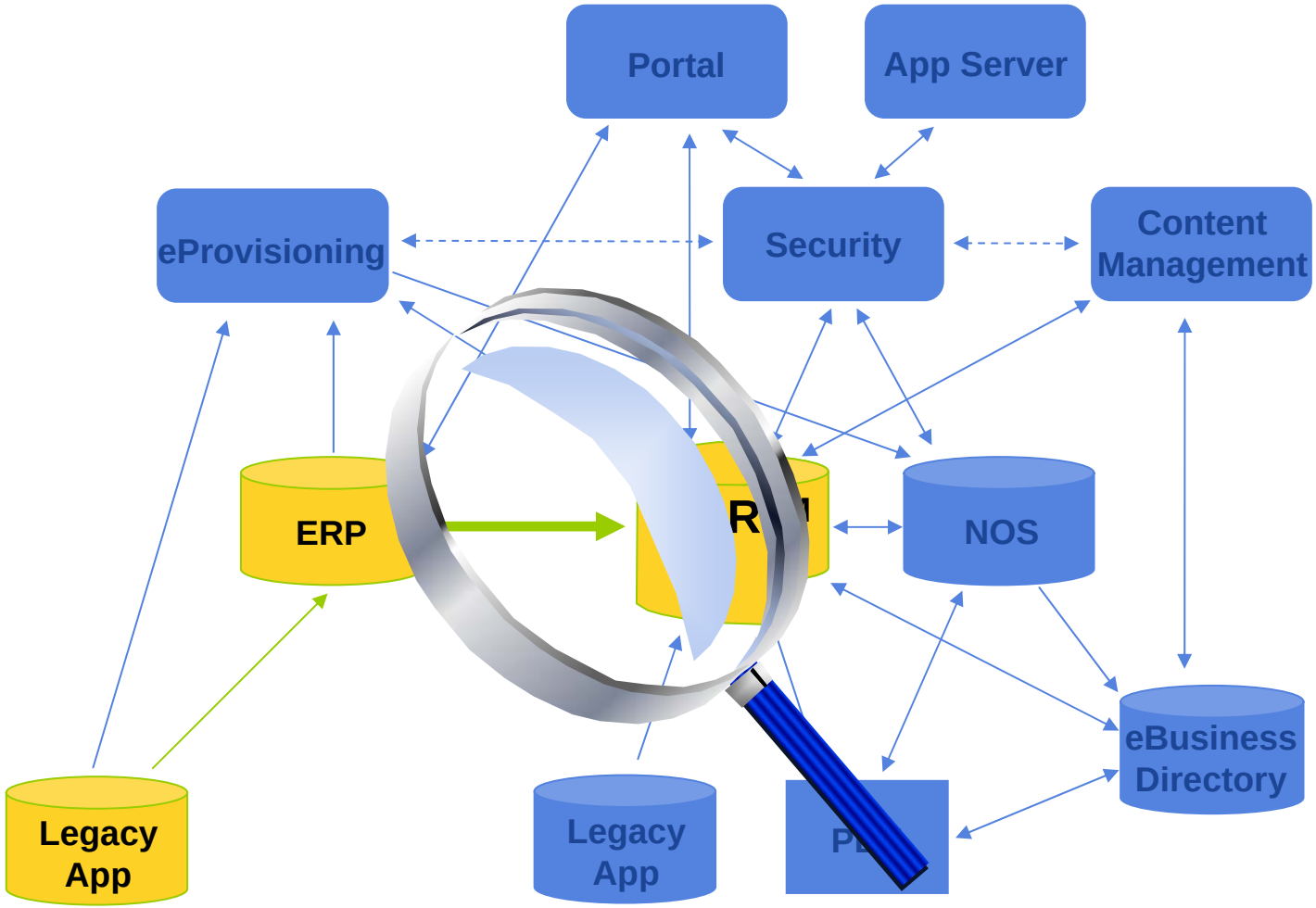
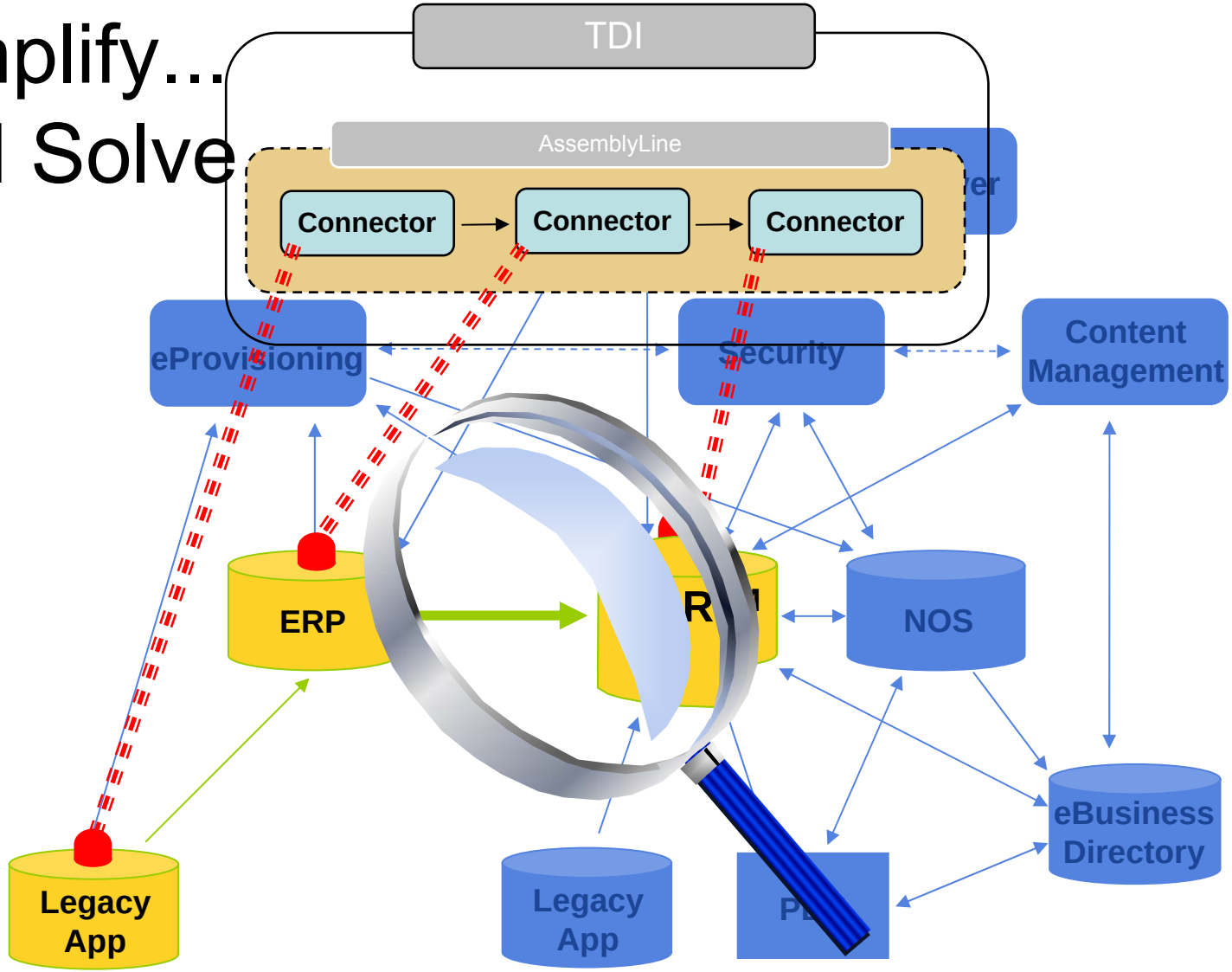

IBM
Christian Châteauvieux
Synchronisez et échangez vos données d'identité entre des applications ou référentiels multiples !
TENDANCES LOGICIELLES D'ÉTÉ 2008
SESSION SPÉCIALE GESTION DES RISQUES OPÉRATIONNELS

Simplify...


Portal
App Server
eProvisioning
Security
Content Management
ERP
NOS
eBusiness Directory
Legacy App
Legacy App

Simplify… and Solve



Central TDI concepts

- *AssemblyLine (AL)*
  - *The ”unit of work” in TDI. All data movement/massaging is done by ALs.*
  - *Activates any number of components, one-by-one, in sequence from top-down*
    - *For example, Script components with JavaScript code that is executed,*
    - *Connectors where the selected Mode logic is carried out (like Update or Lookup),*
    - *Branches and Loops used to articulate flow logic in the AL*
  - *May implement multiple Operations, like SyncDBfromSource does (add, modify & delete)*

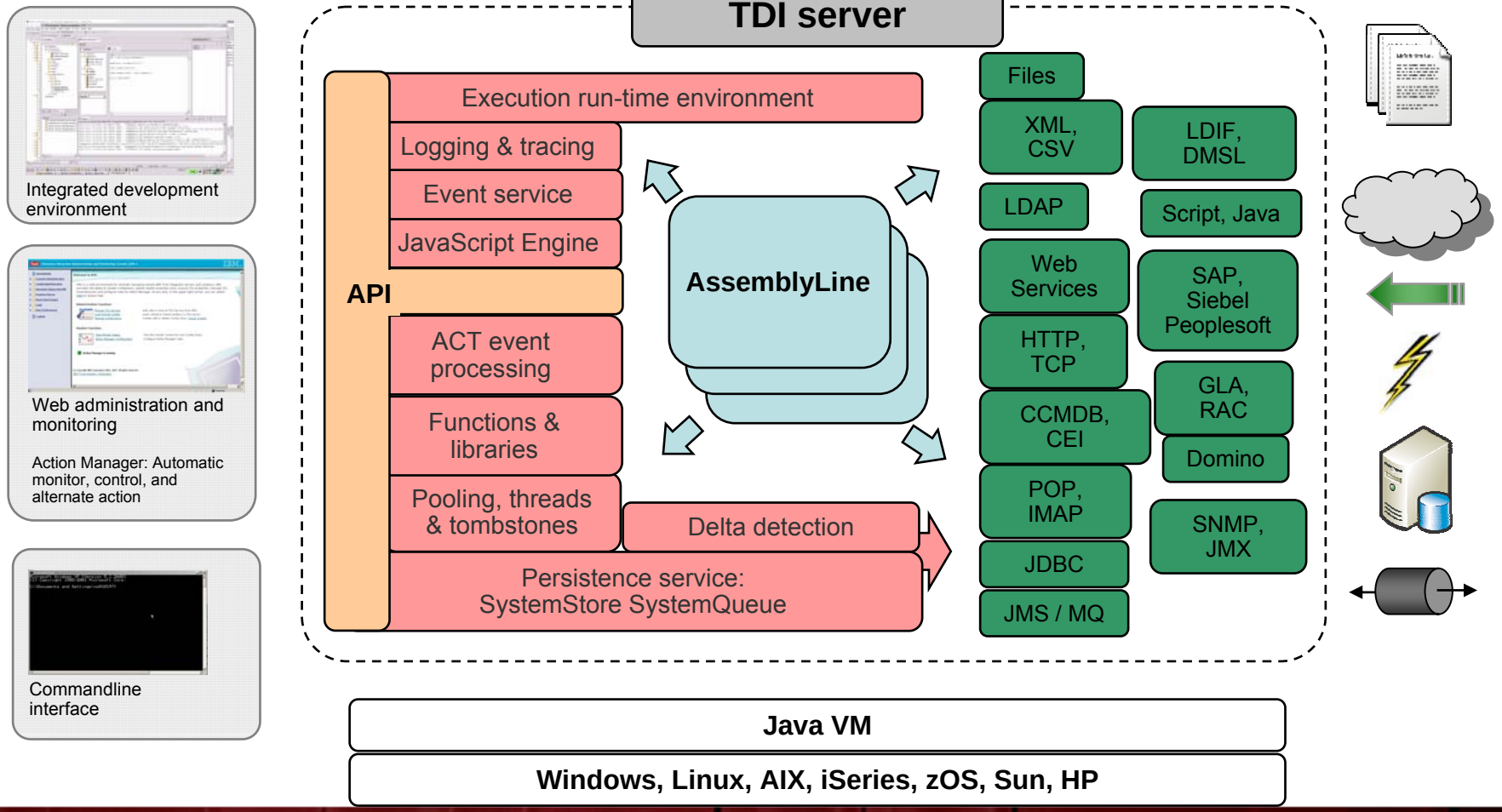
- *TDI Components*
  - *Connector* — *provides a connection to some external data source; may require a Parser (like FileSystem or MemoryStream); behaves according to its Mode setting, eg. AddOnly, Iterator, Lookup, Delete & Update; offers Input & Output Map (Mode dependant)*
  - *Script* — *contains JavaScript code to execute*
  - *Function (FC)* — *perform some function, like calling a web service or sending email; offers Output Map (call parameters) & Input Map (return values)*
  - *AttMap* — *Attribute map from work to work (conn not involved)*
  - *Branch, Switch, Loop* — *flow control components*

Kernel/Component architecture

- Kernel accelerates development through:
  - automated data pipeline behaviors
  - integrated scripting framework
  - solution deployment features & tools:
    - *log-files, tracing*
    - *error handling*
    - *security*
    - *management interface/API*
    - ...

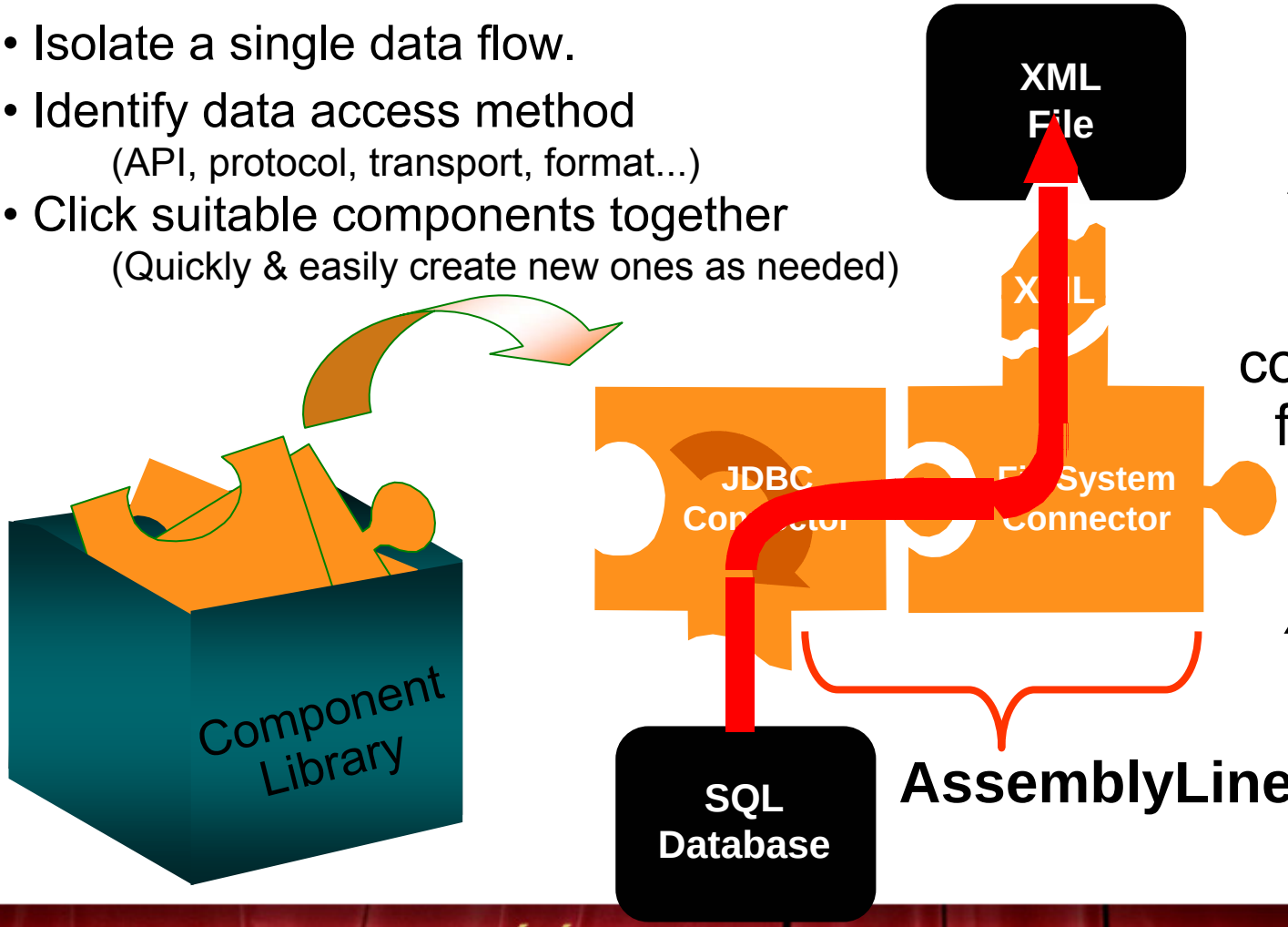
- Components create a 'level playing field' by abstracting away the details of:
  - transports
  - APIs
  - protocols
  - data/file formats

TDI architecture



Rapid Integration Development

- Isolate a single data flow.
- Identify data access method
  (API, protocol, transport, format...)
- Click suitable components together
  (Quickly & easily create new ones as needed)



A collection of components that form a continuous path from source(s) to target(s) is called an *AssemblyLine*

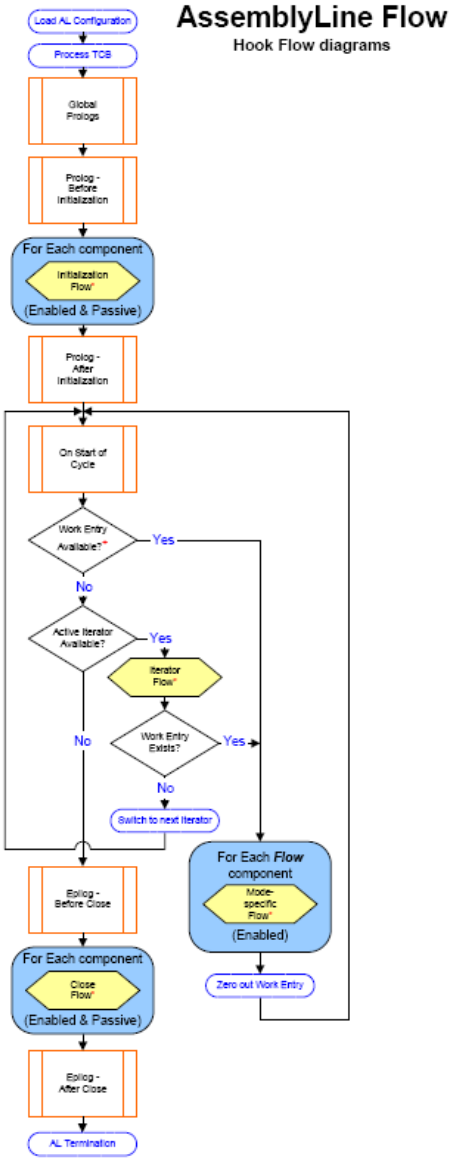
Data Pipeline

- AssemblyLine
  - Fixed flow logic for driving components in the specified order
  - User-defined flow logic through component ordering
- Components
  - All components have a fixed flow logic, except Scripts and AttMaps
  - Connector flow determined by Mode setting
- Error/Exception Flows
  - Connection Failure (& Auto-Reconnect feature)
  - Exceptional situations (e.g. Lookup finds none or multiple entries)
  - Error Handling
- User-defined (JavaScript)

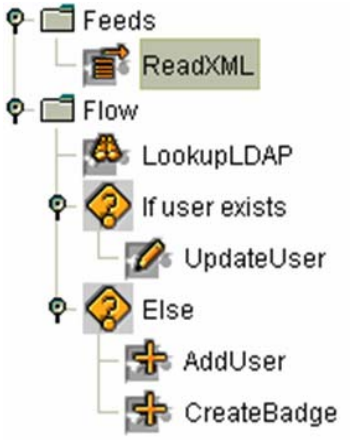
DataFlow Diagrams:http://publib.boulder.ibm.com/infocenter/tivihelp/v2r1/topic/com.ibm.IBMDI.doc_6.1.1/TDI_6.1_FlowDiagrams.pdf

AssemblyLine Flow

The built-in pipeline drives the Assemblyline, allowing default behavior to be augmented or even overridden wherever needed by scripting in *Hooks.*

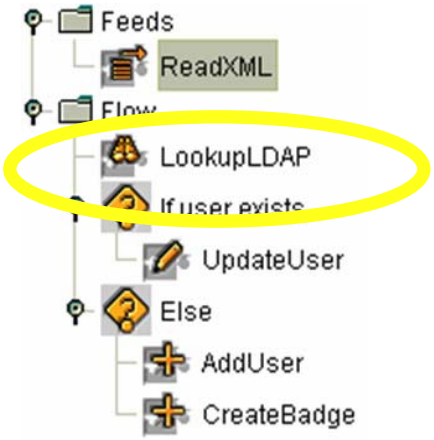


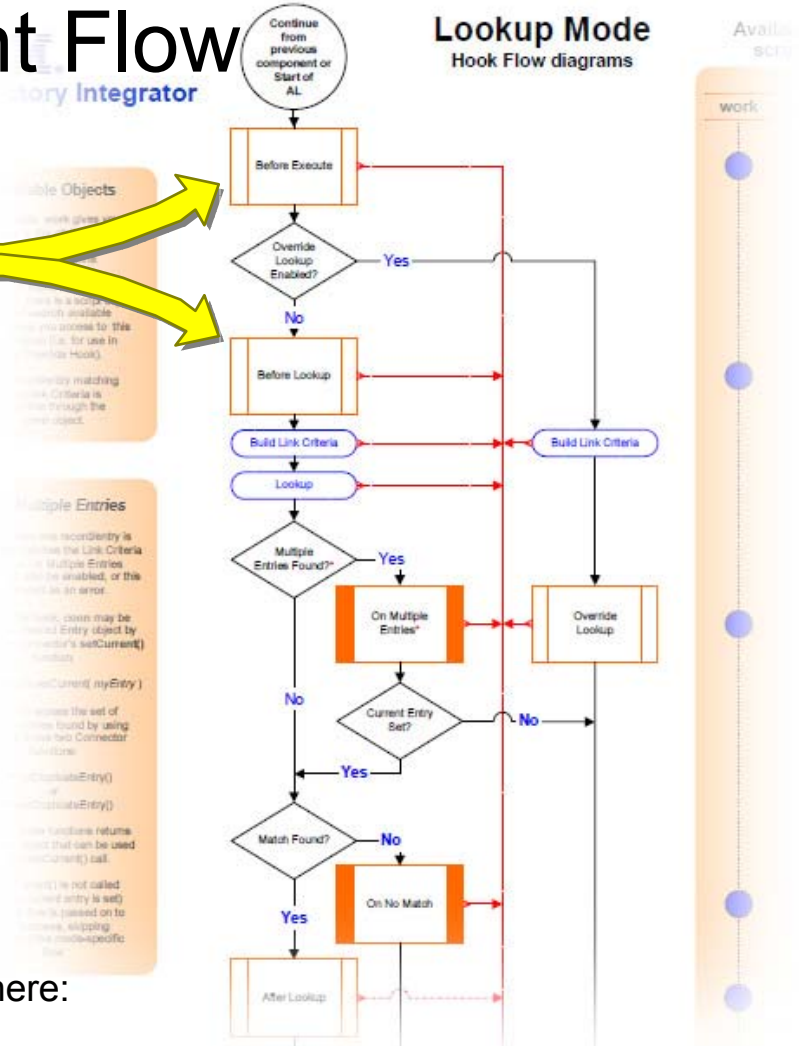
The AssemblyLine visualizes custom data flows in the graphical development UI. Each component is configurable and is controlled by additional pipeline models



Component Flow

The component pipelines specific to the role of each component, and just like the AL flow are also configurable and can either be enhanced or overridden by scripting *Hooks.*





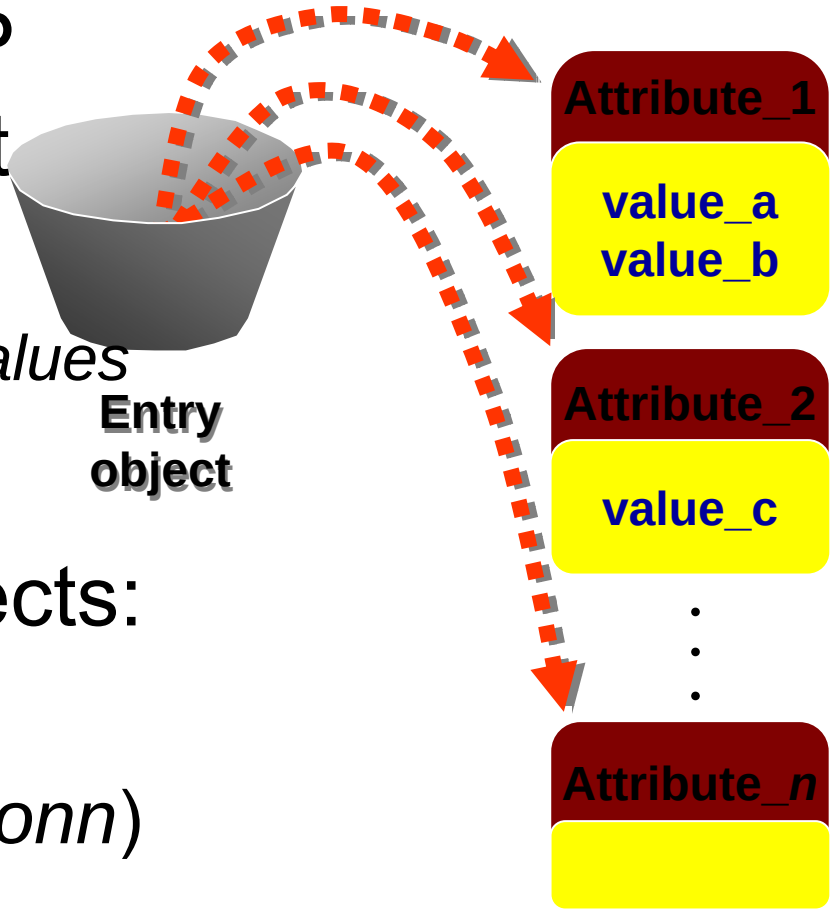
The comprehensive list of Flow diagrams are found here:

http://publib.boulder.ibm.com/infocenter/tivihelp/v2r1/topic/com.ibm.IBMDI.doc_6.1.1/TDI_6.1_FlowDiagrams.pdf

TDI Entry-Attribute-value data model

- Similar to how LDAP directories represent data:

  entries ➲ attributes ➲ values

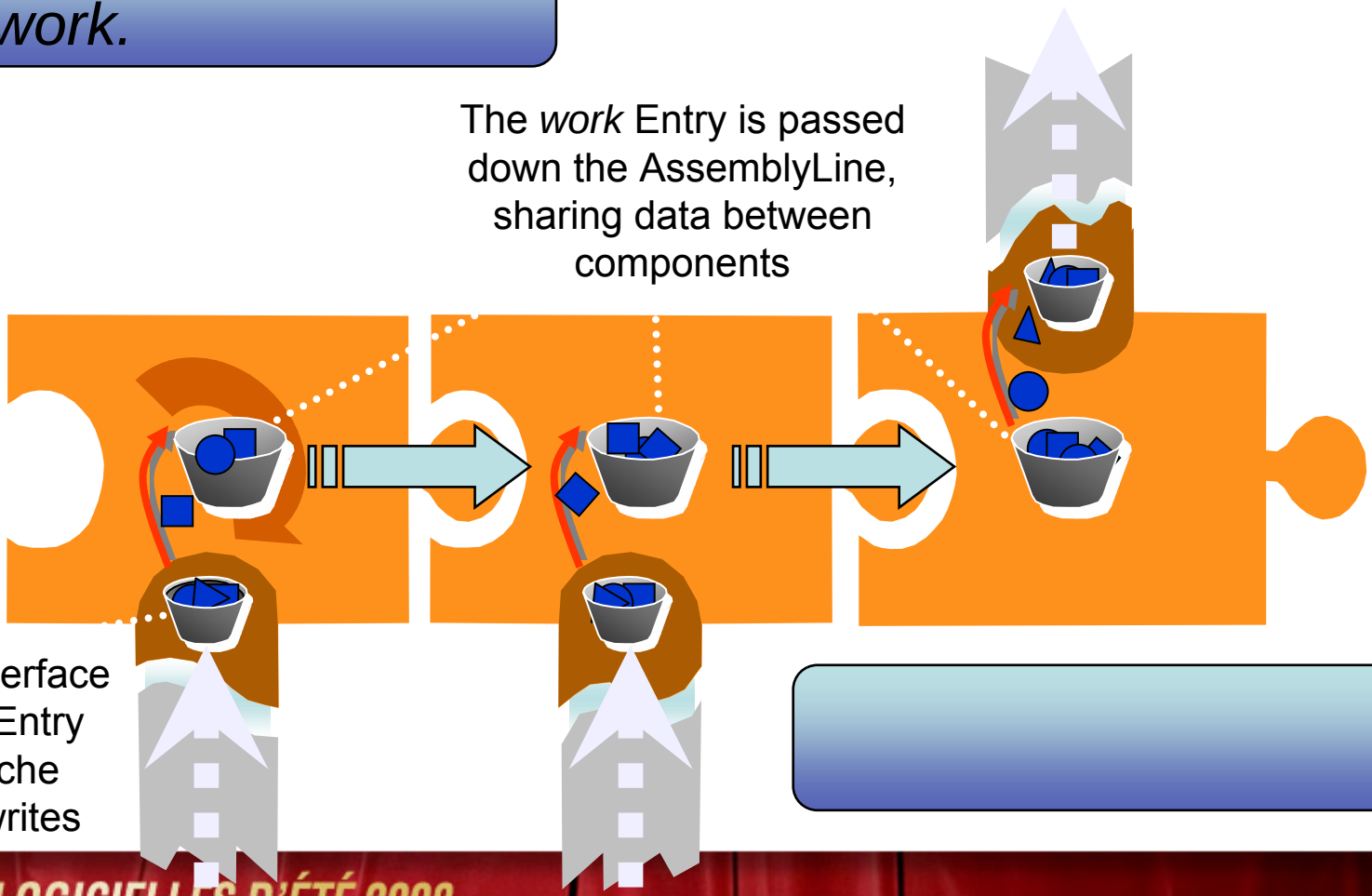


- Two main Entry objects:
  1. Work Entry (*work*)
  2. Connector Entry (*conn*)

AssemblyLine cycling

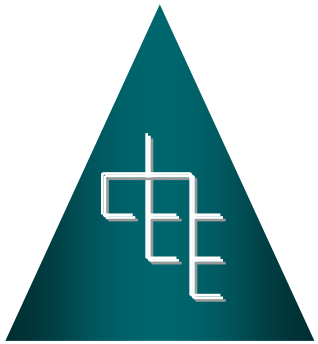
Input Maps moves data from *conn* to *work.*

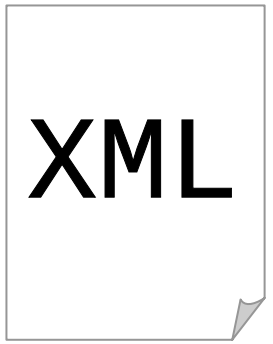
The *work* Entry is passed down the AssemblyLine, sharing data between components



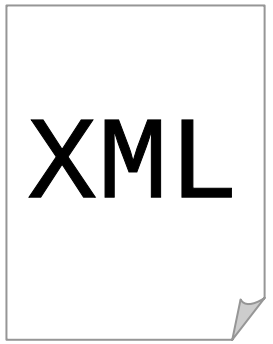
Each Connector Interface has a *conn* Entry used to cache reads and writes

Read DB, Join from LDAP, Write to File

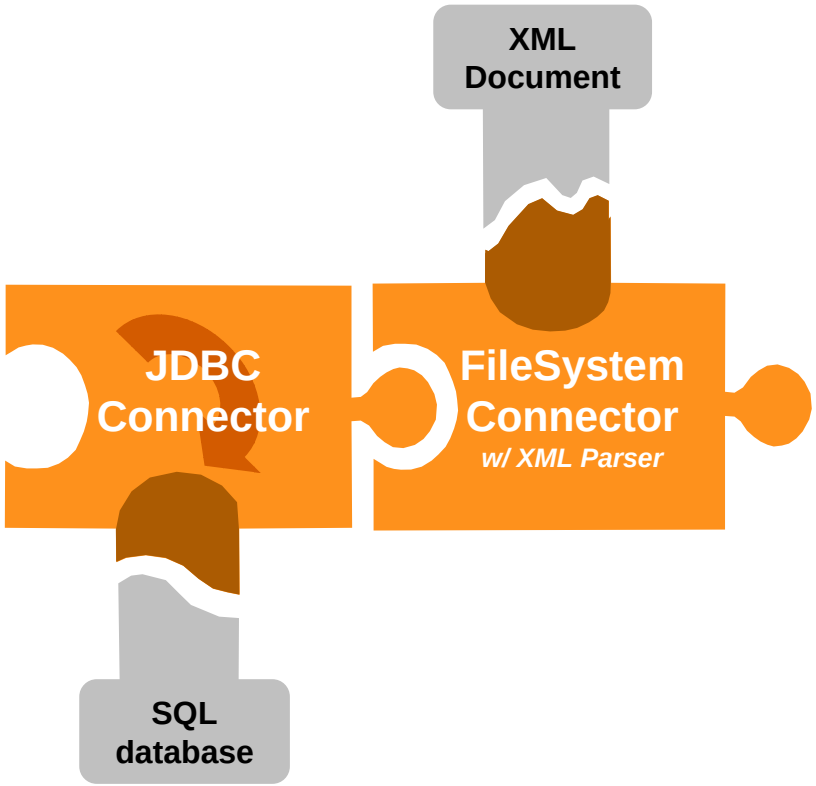




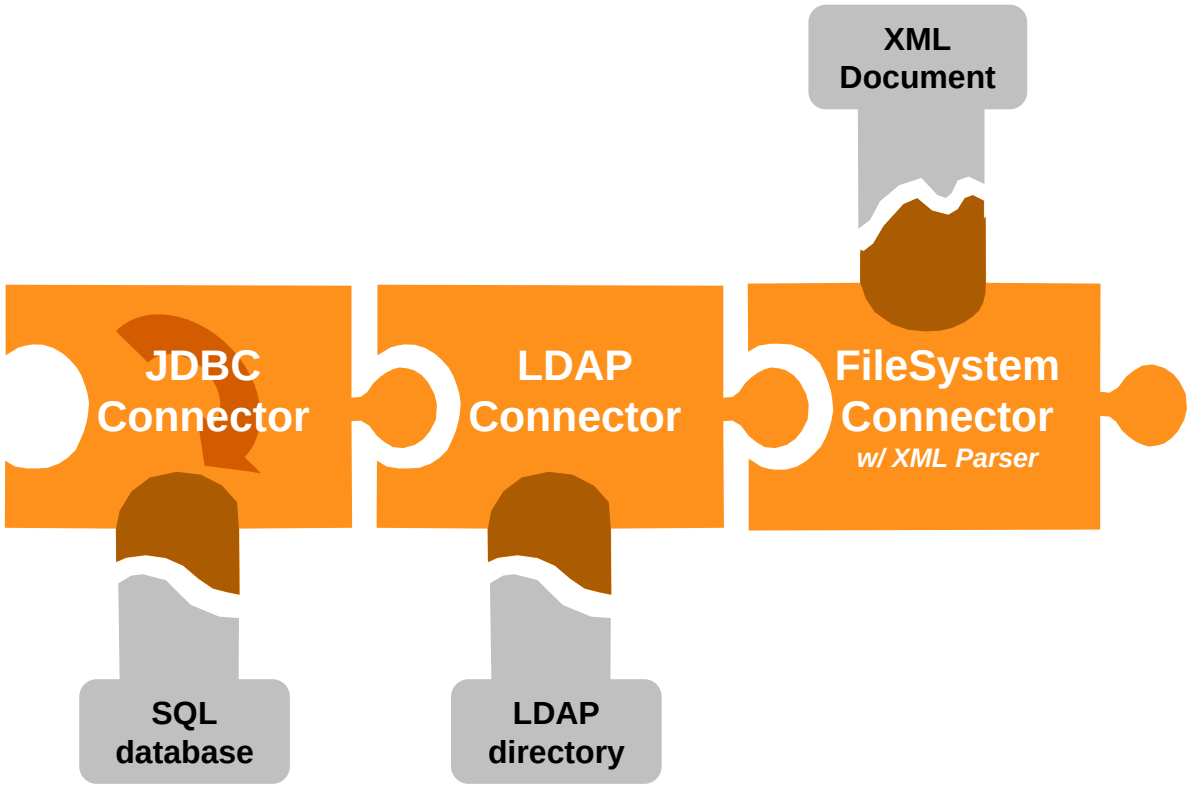
Read DB, Join from LDAP, Write to File



Step1. Read DB, Write to File



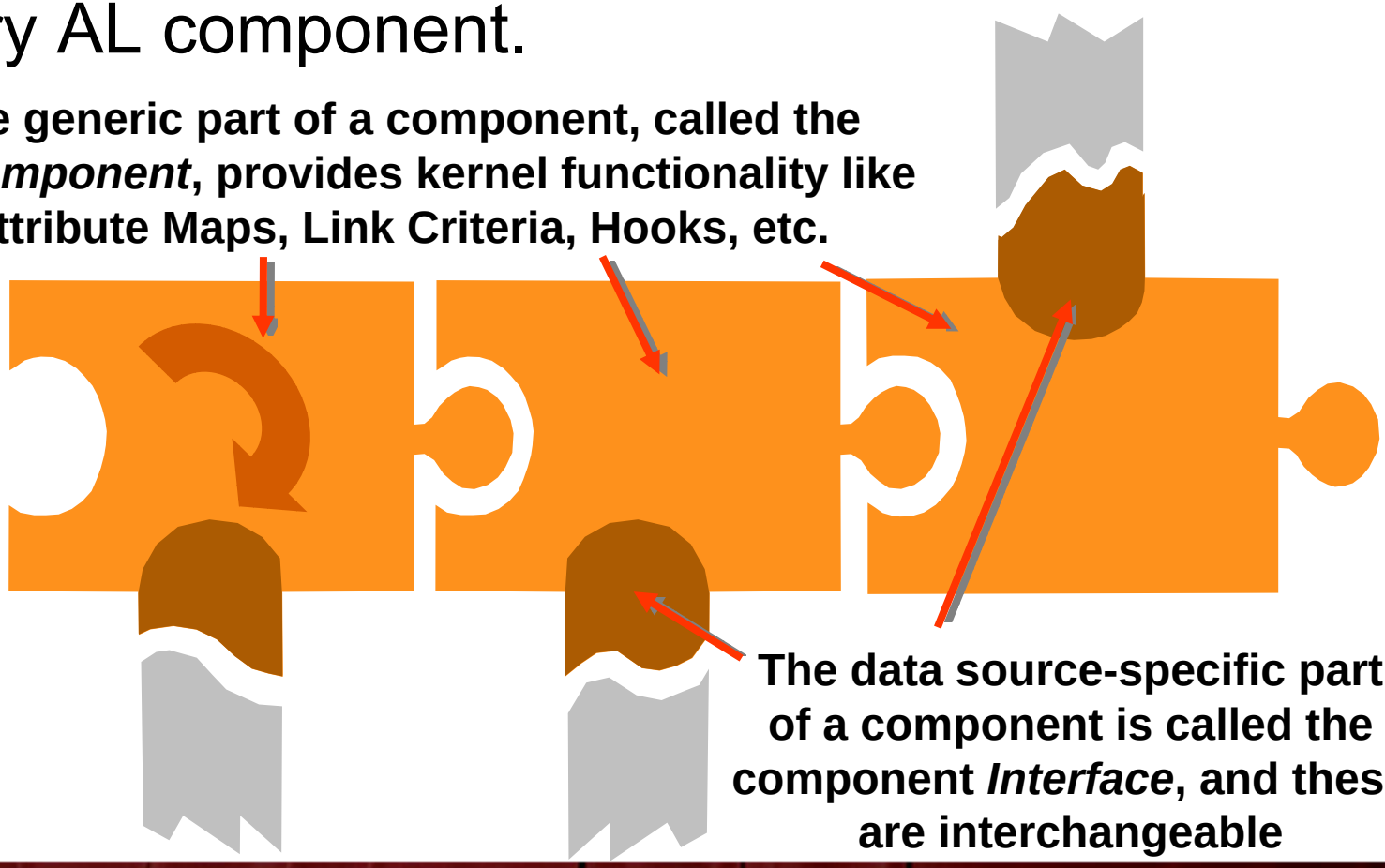
Step 2. Add Join from LDAP



Inside an AssemblyLine

The kernel/component architecture is visible in every AL component.

The generic part of a component, called the *AL component*, provides kernel functionality like Attribute Maps, Link Criteria, Hooks, etc.

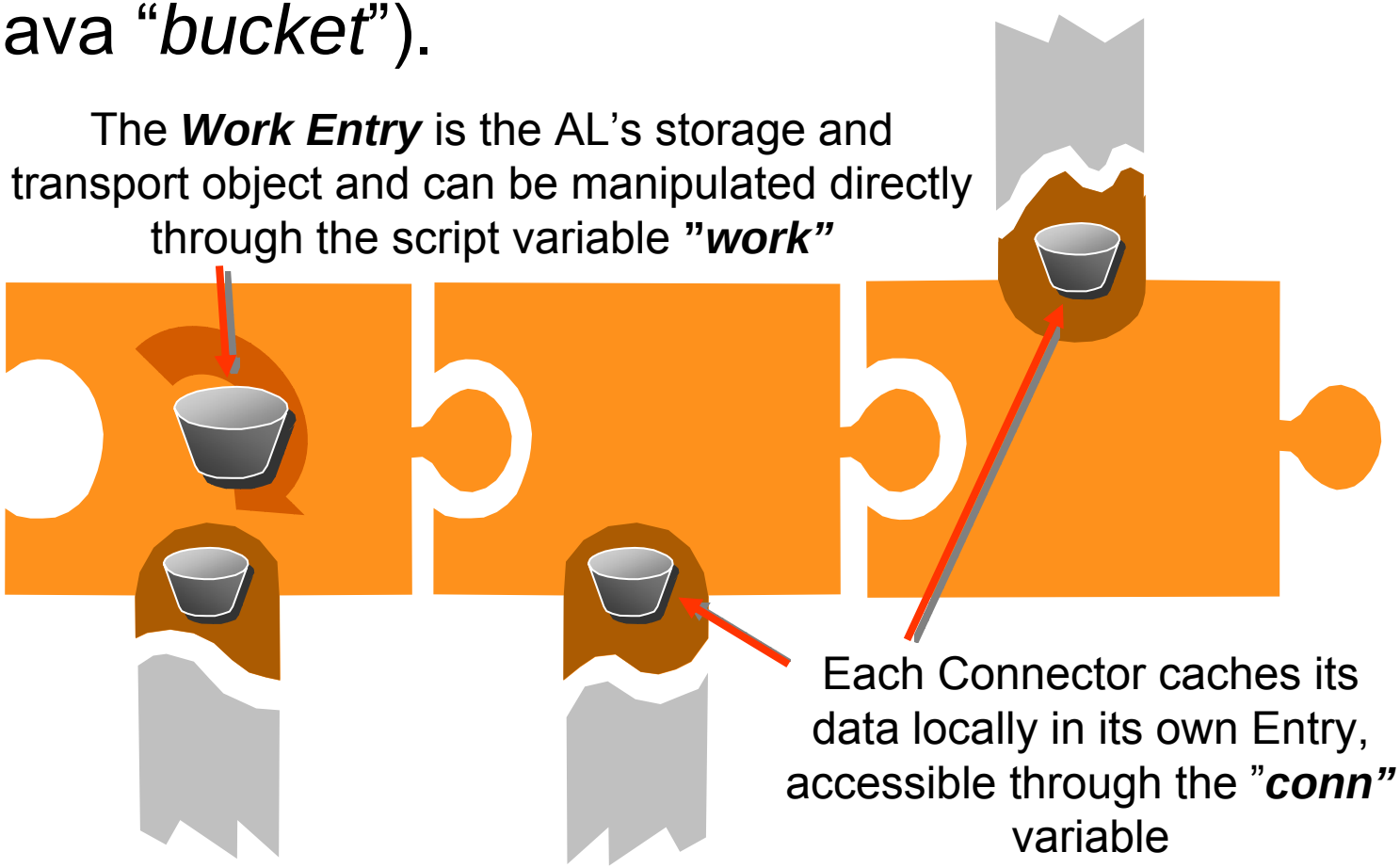


The data source-specific part of a component is called the component *Interface*, and these are interchangeable

Inside an AssemblyLine

Entry objects are used to store and transport the data (Java "*bucket*").

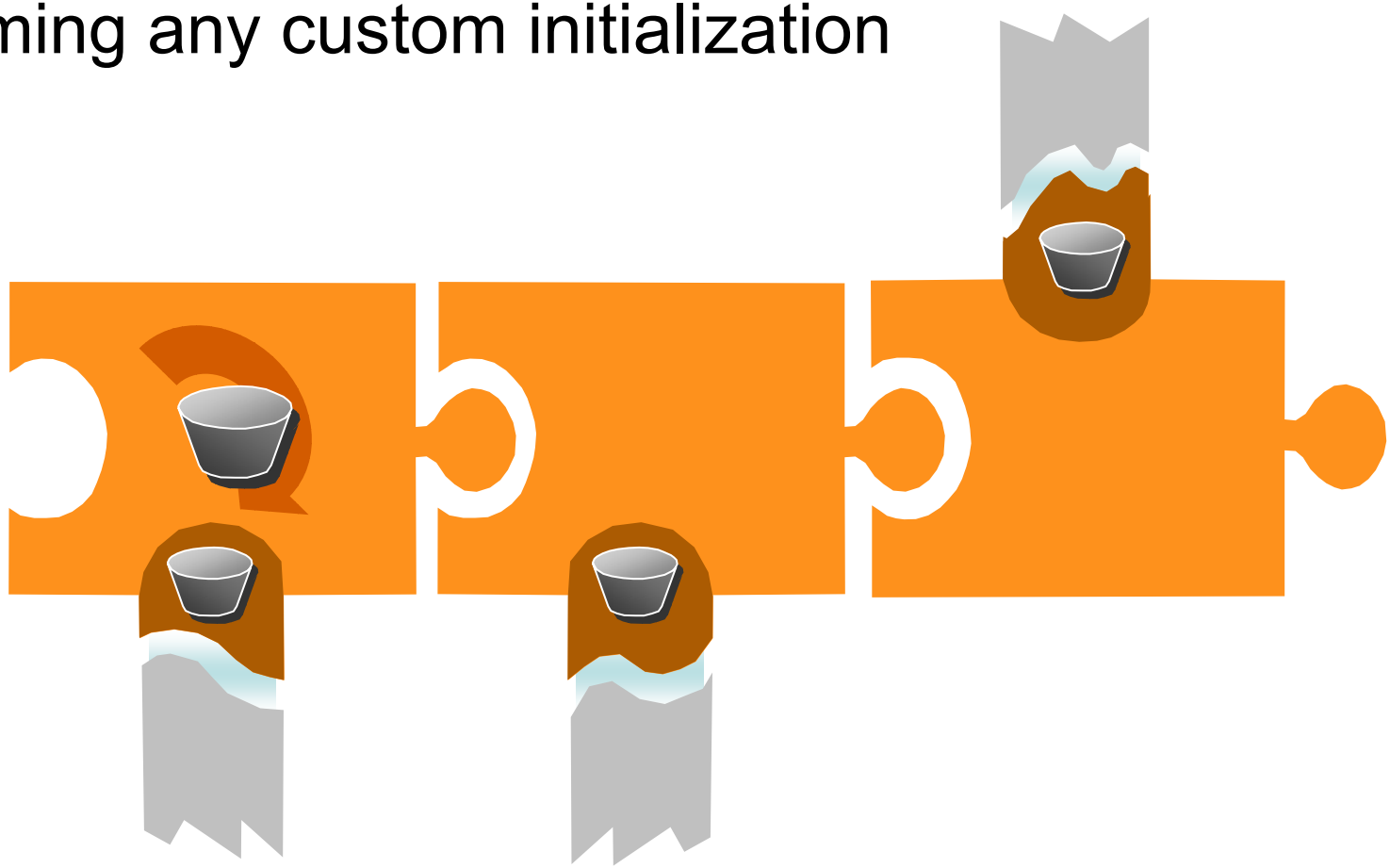
The ***Work Entry*** is the AL's storage and transport object and can be manipulated directly through the script variable ***"work"***



Each Connector caches its data locally in its own Entry, accessible through the "***conn"*** variable

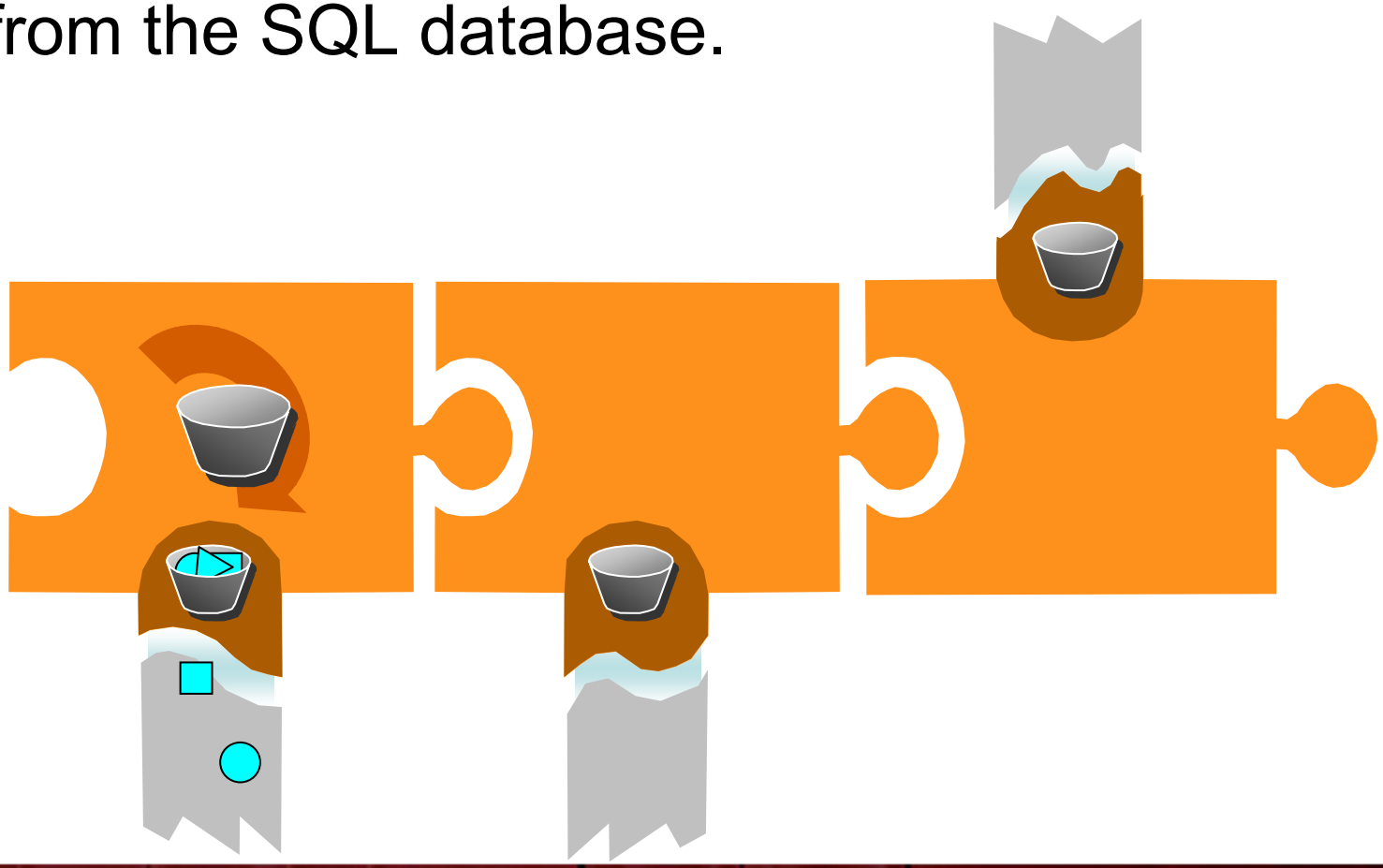
Inside an AssemblyLine

When the AL starts, it fires up all components, performing any custom initialization logic.



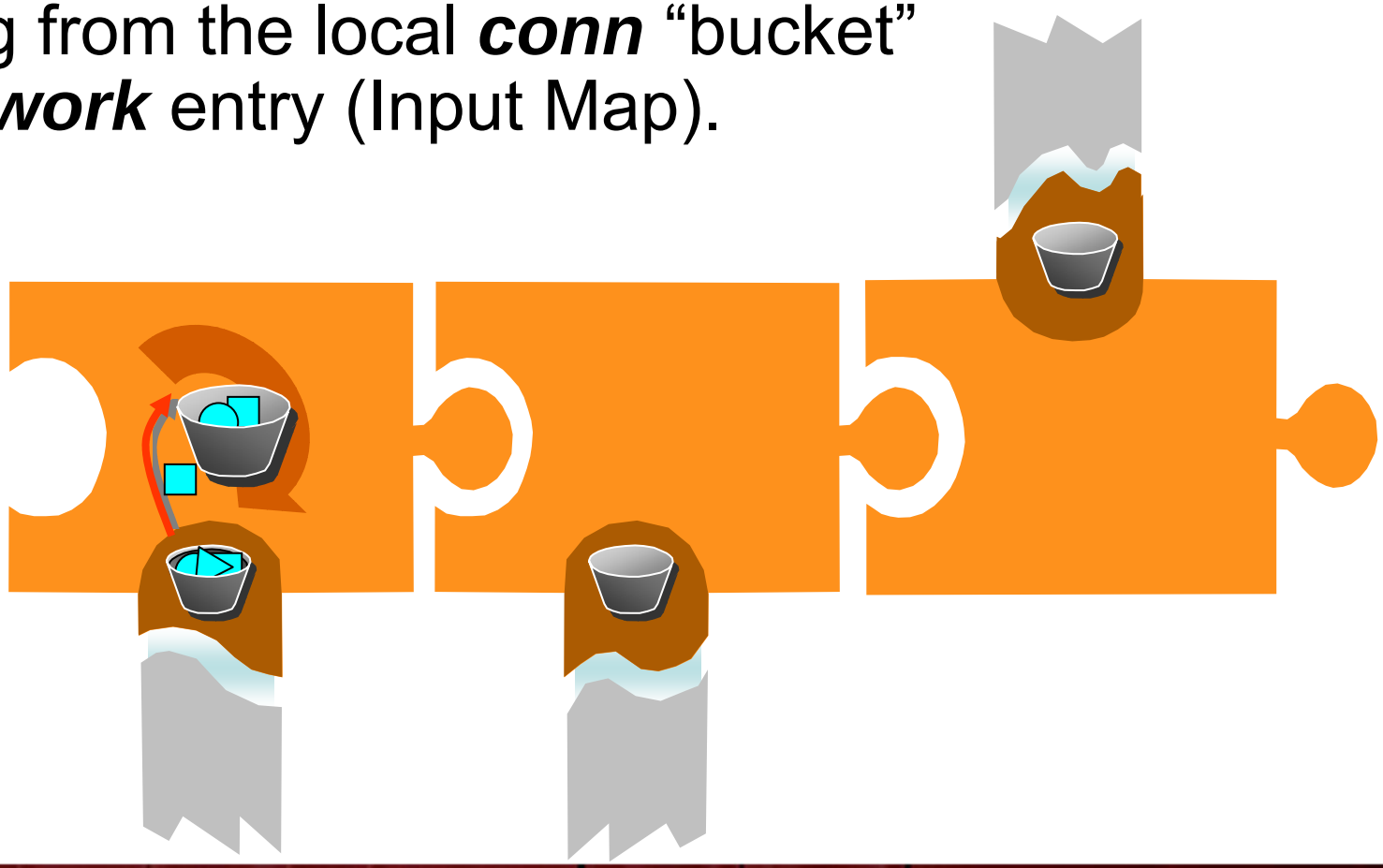
Inside an AssemblyLine

The first component (Iterator Connector) reads a row from the SQL database.



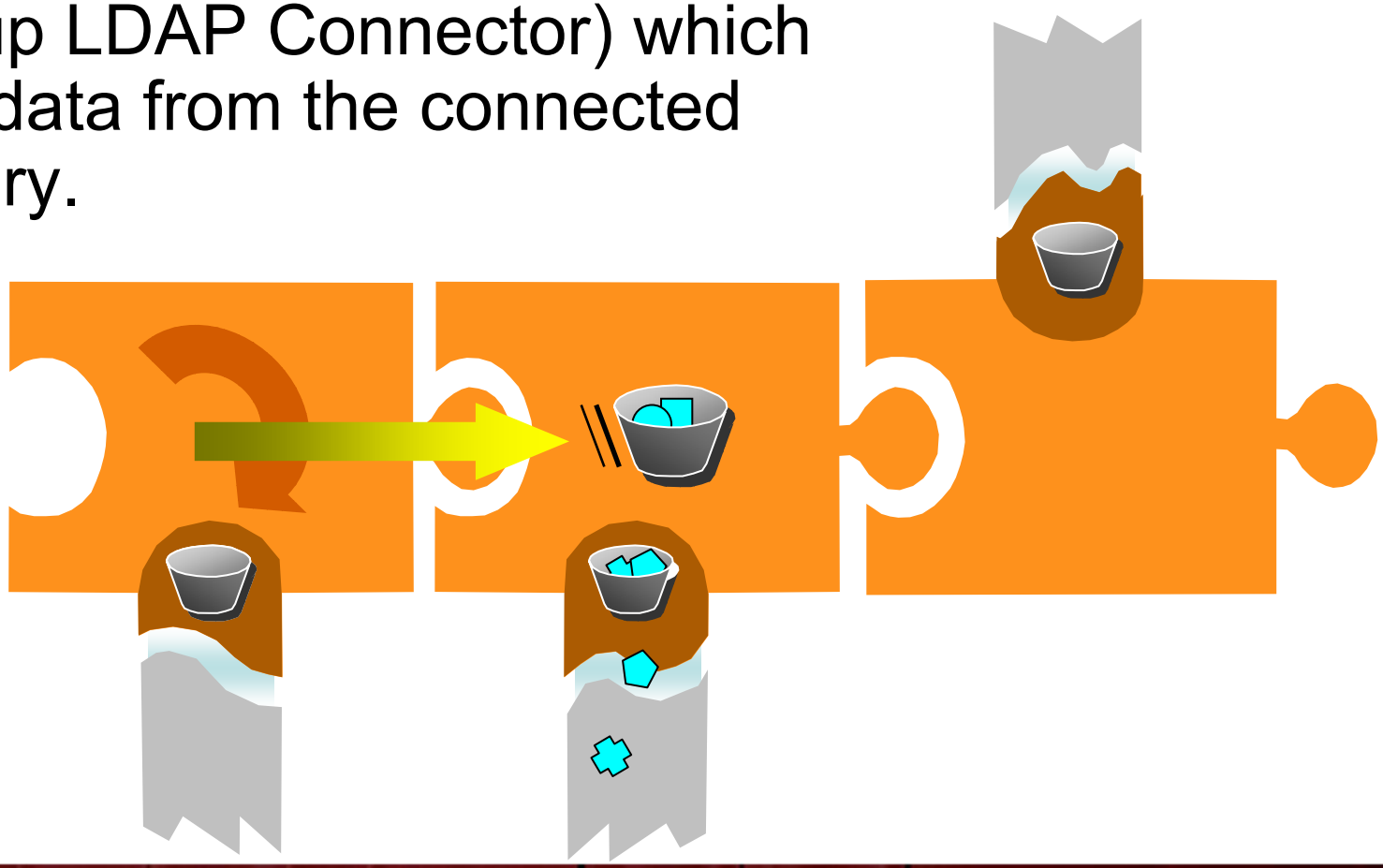
Inside an AssemblyLine

Selected attributes are mapped into the flow, moving from the local ***conn*** "bucket" to the ***work*** entry (Input Map).



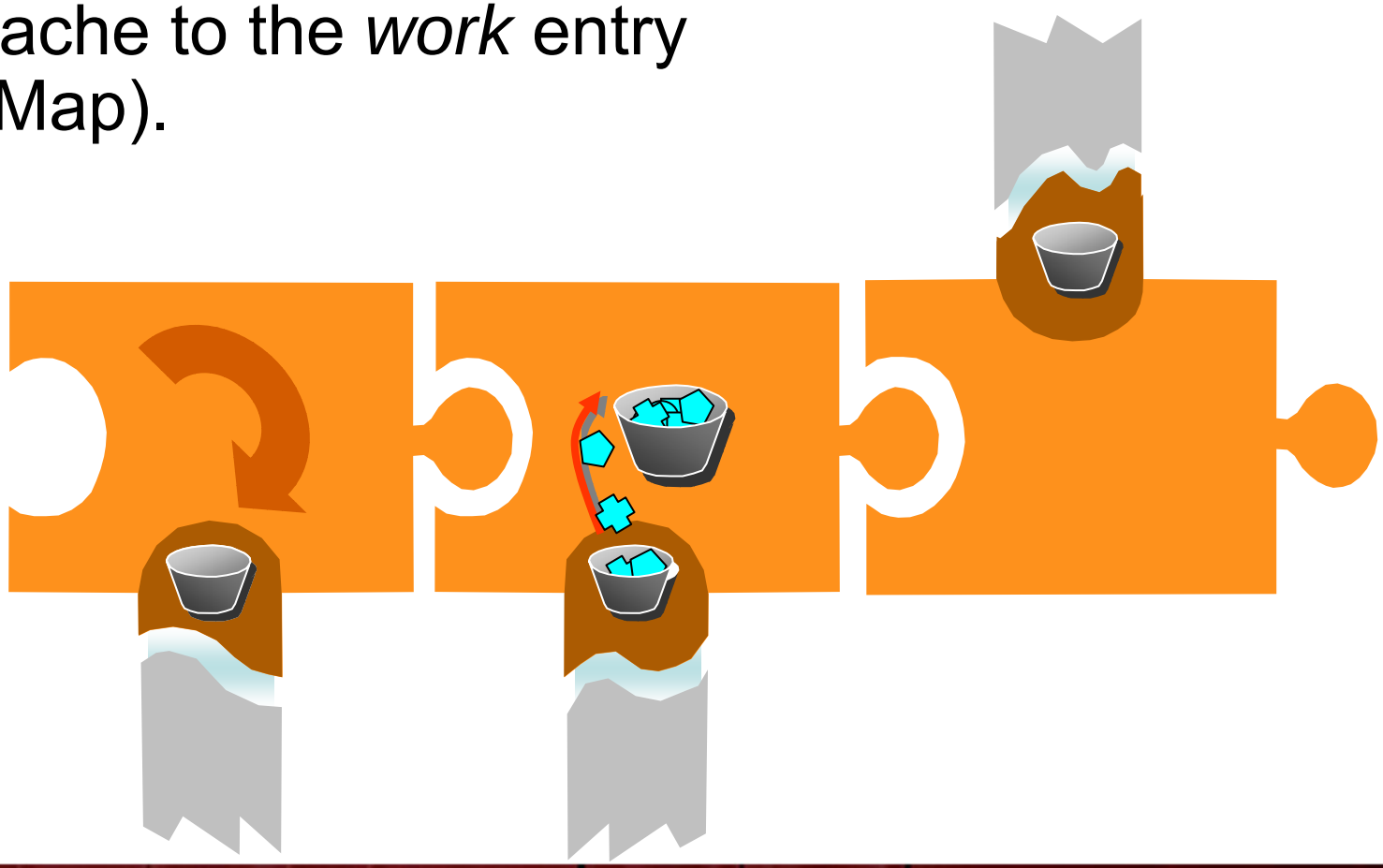
Inside an AssemblyLine

The *work* entry moves to the next component (Lookup LDAP Connector) which reads data from the connected directory.



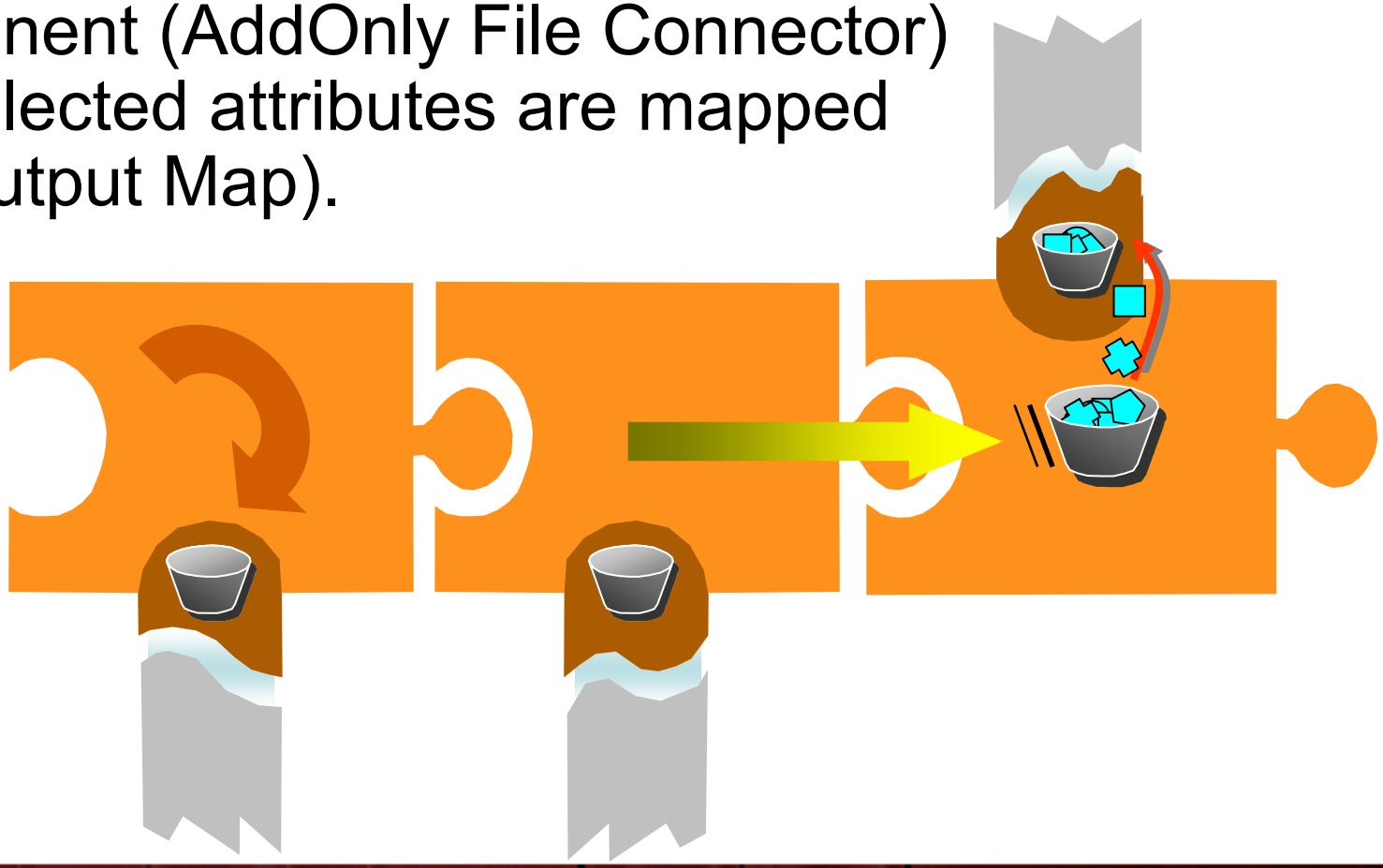
Inside an AssemblyLine

More attributes are mapped from the local *conn* cache to the *work* entry (Input Map).



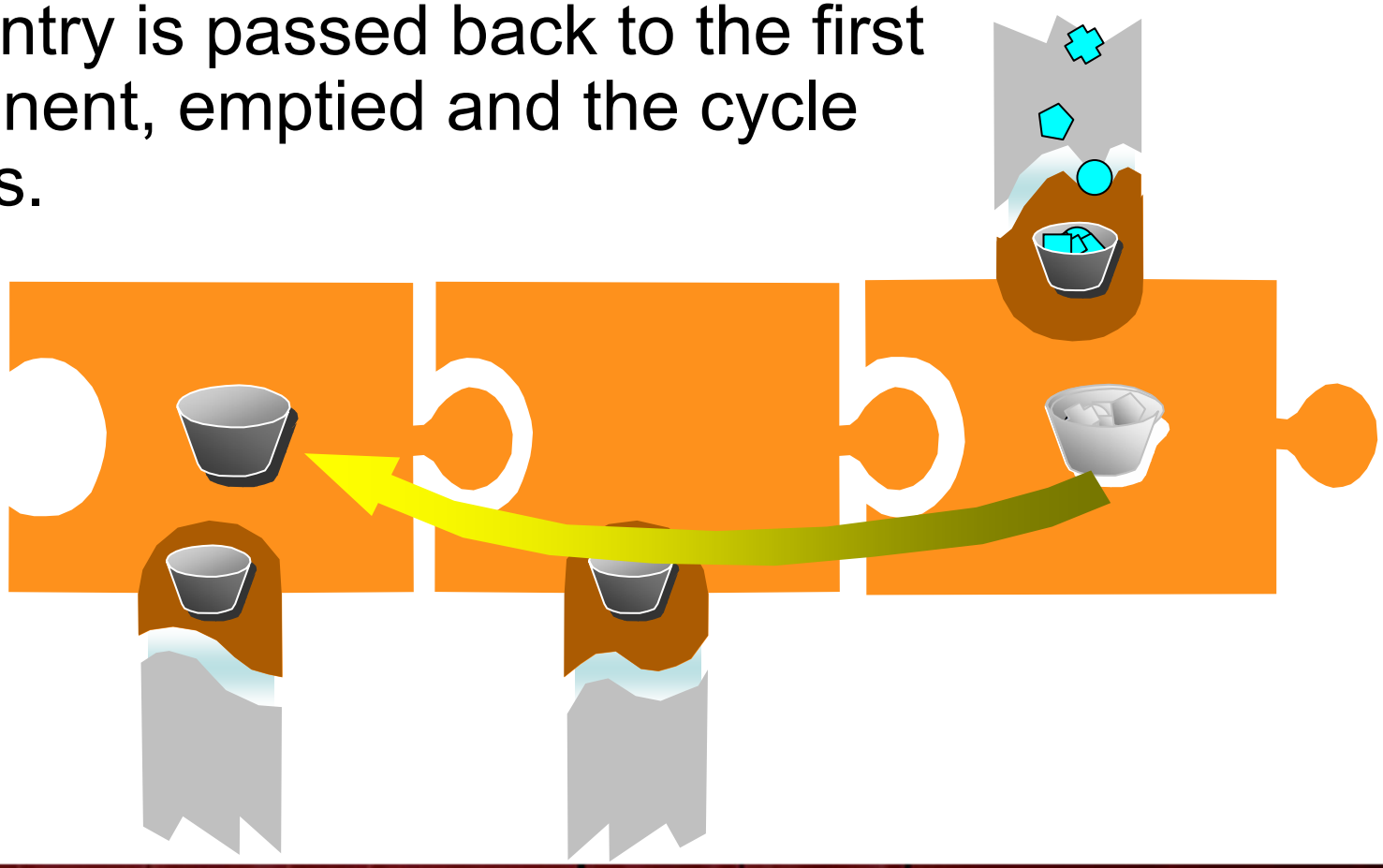
Inside an AssemblyLine

Now the *work* entry is passed to the last component (AddOnly File Connector) and selected attributes are mapped out (Output Map).



Inside an AssemblyLine

Finally, the output operation is performed. The *work* entry is passed back to the first component, emptied and the cycle repeats.



TDI 6.1.1 Server

Axis Easy Web Service Server Connector
Axis Easy Web Service Invoke
Axis Java-to-Soap
Invoke Soap Web Service
Axis Soap-to-Java
Complex Types Generator
Wrap Soap

LDAP Connector
LDAP Server Connector
Tivoli Access Manager Connector
Windows Users and Groups Connector

Active Directory Changelog Connector v2
IBM Directory Server Changelog Connector
Netscape/iPlanet Changelog Connector
zOS LDAP Changelog Connector

BTree Connector
JDBC Connector
Properties Connector
SystemStore Connector
RDBMS Changelog Connector

AssemblyLine Connector
Server Notifications Connector
AssemblyLine Function Component

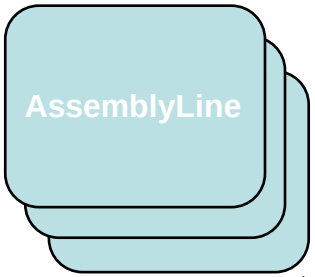
Domino Change Detection Connector
Domino Users Connector
Lotus Notes Connector

Exchange Changelog Connector
Mailbox Connector
SendEMail Function Component

TIM DSMLv2 Connector
DSMLv2 SOAP Connector
DSML v2 SOAP Server Connector
Generic JNDI Connector
ITIM Agent Connector

EMF SDOToXML Function Component
EMF XMLToSDO Function Component

Timer Connector



Active Correlation Technology Connector
Generic Log Adapter Connector
RAC Connector
Entry to CommonBaseEvent Function

JMX Connector
SNMP Connector
SNMP Server Connector
TCP Connector
TCP Server Connector

Remedy/Peregrine /CCMDB tickets

Many Custom Components downloadable
from OPAL or tdi-users.org or on request

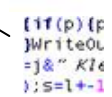
CCMDB

Netcool

RSS

Google

PeopleSoft Connector
Siebel Connector
SAP ALE IDoc Connector
SAP R/3 Business Object Repository
SAP R/3 User Registry
SAP R/3 RFC Functional Component



Script Connector
Generic Java Method
Parser FC
Scripted Function Component

Remote Command Line Function Component
z/OS TSO/E Command Line Function Component
Command Line Connector

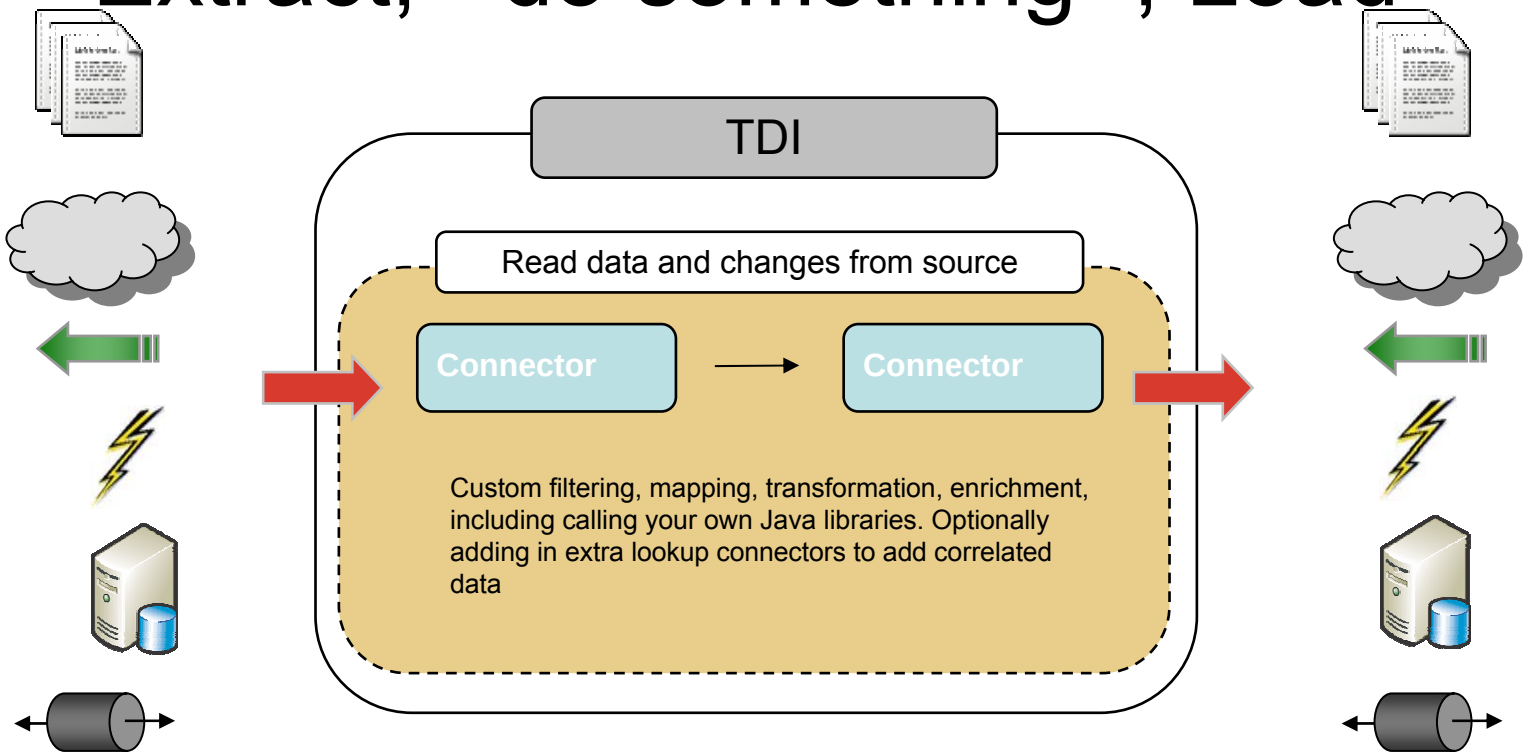
Memory Queue FC
MemQ Connector
Memory Stream Connector

File System Connector
FTP Client Connector
URL Connector
HTTP Client
HTTP Server Connector

IBM MQ Series Connector
JMS Pub/Sub Connector
MQe Password Store Connector
System Queue Connector

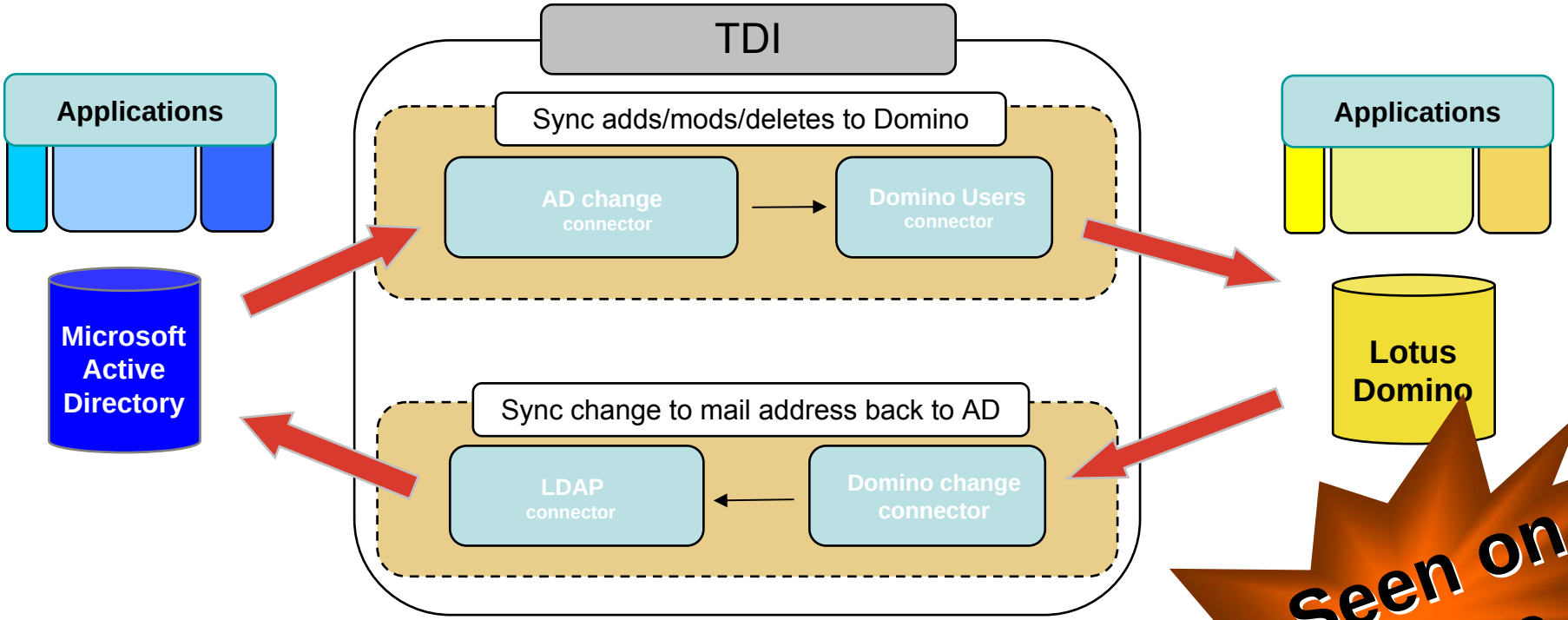
CSV Parser
DSML v1 Parser
DSML v2 Parser
Fixed Record Parser
HTTP Parser
LDIF Parser
Line Reader/Writer
SOAP Parser
Script Parser
Simple Parser
XML Parser
XML Sax Parser
XSL based XML Parser

Extract, <do something>, Load



TDI has the capability to detect changes in the source data each time it reads from it. For example, an HR report might not contain information about changes, only the entire snapshot of the entire employee database. TDI is able to determine the change from previous versions and only propagate added, changed and deleted records

Directory/Datastore Synchronization

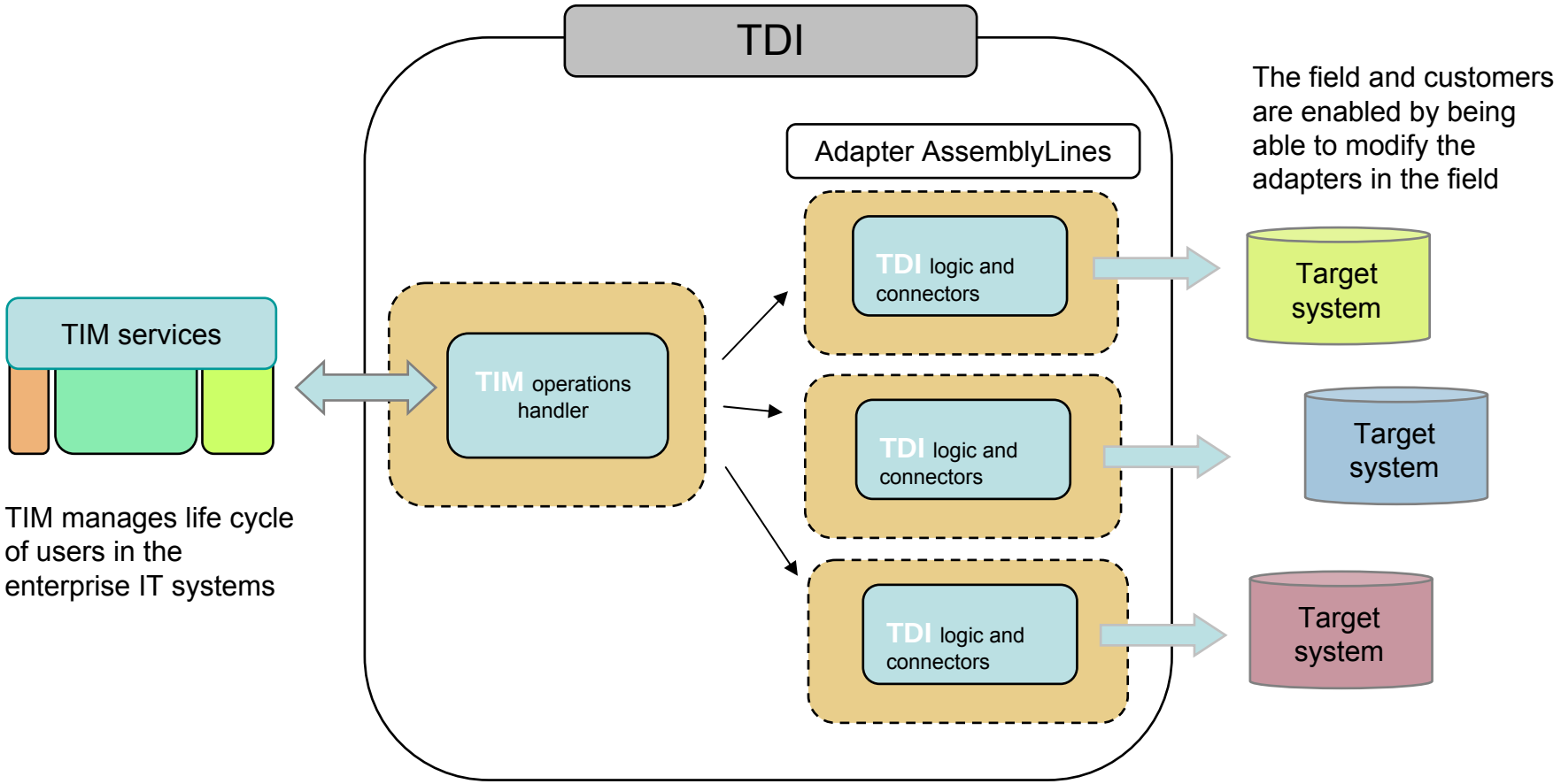


In this scenario, users are managed in AD and need to be synchronized with Domi

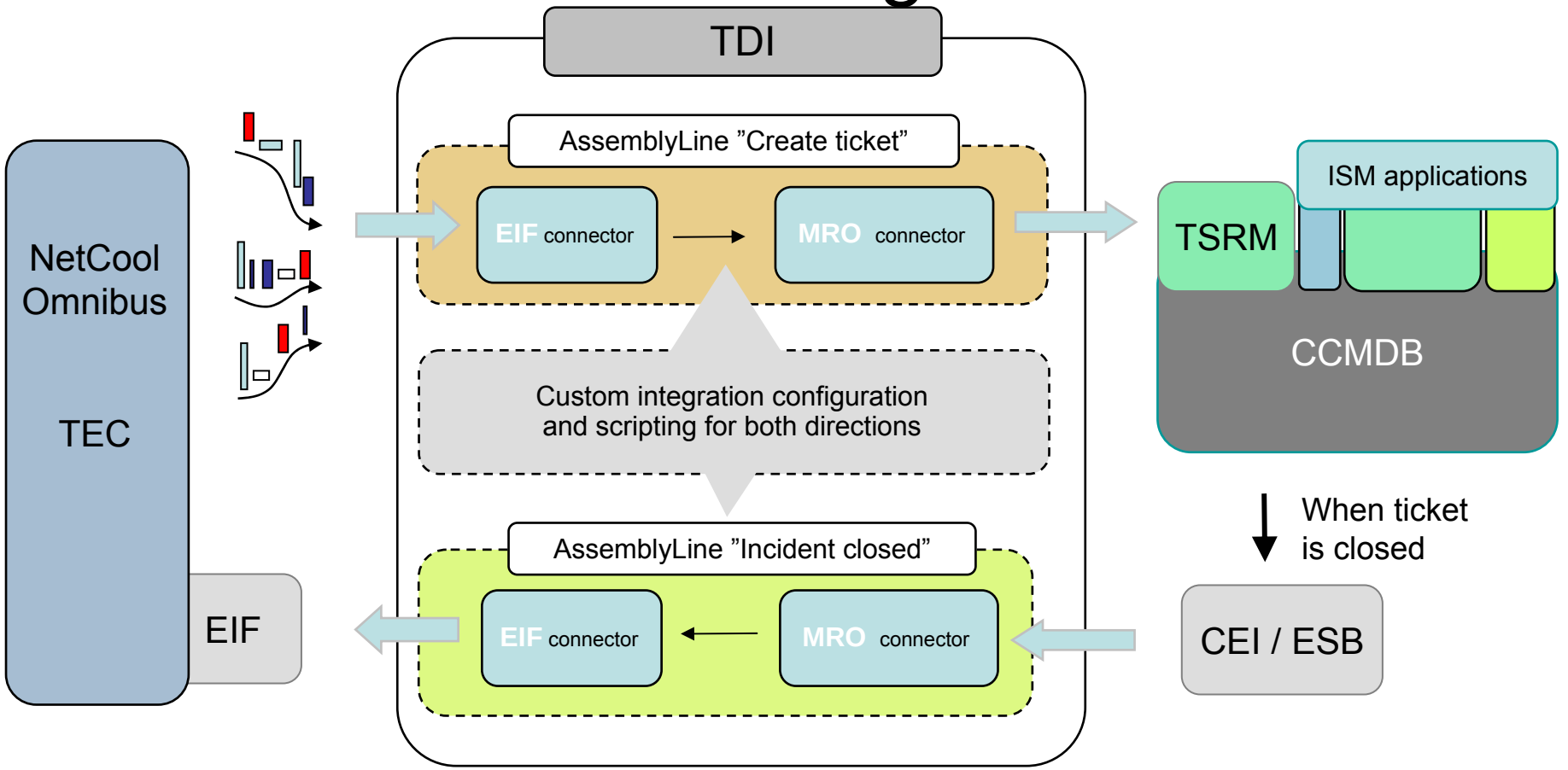
1. All data is pushed into Domino when new Users are added in AD.
2. E-mail is ”owned” by Domino, so changes in AD will not overwrite the ”Int ino.
3. Changes to the internet mail address in Domino are sync’ed to AD.
4. Any of the above logic can be easily and quickly modified and extended.

As Seen on YouTube

Adapter framework for ITIM (Tivoli Identity Manager)

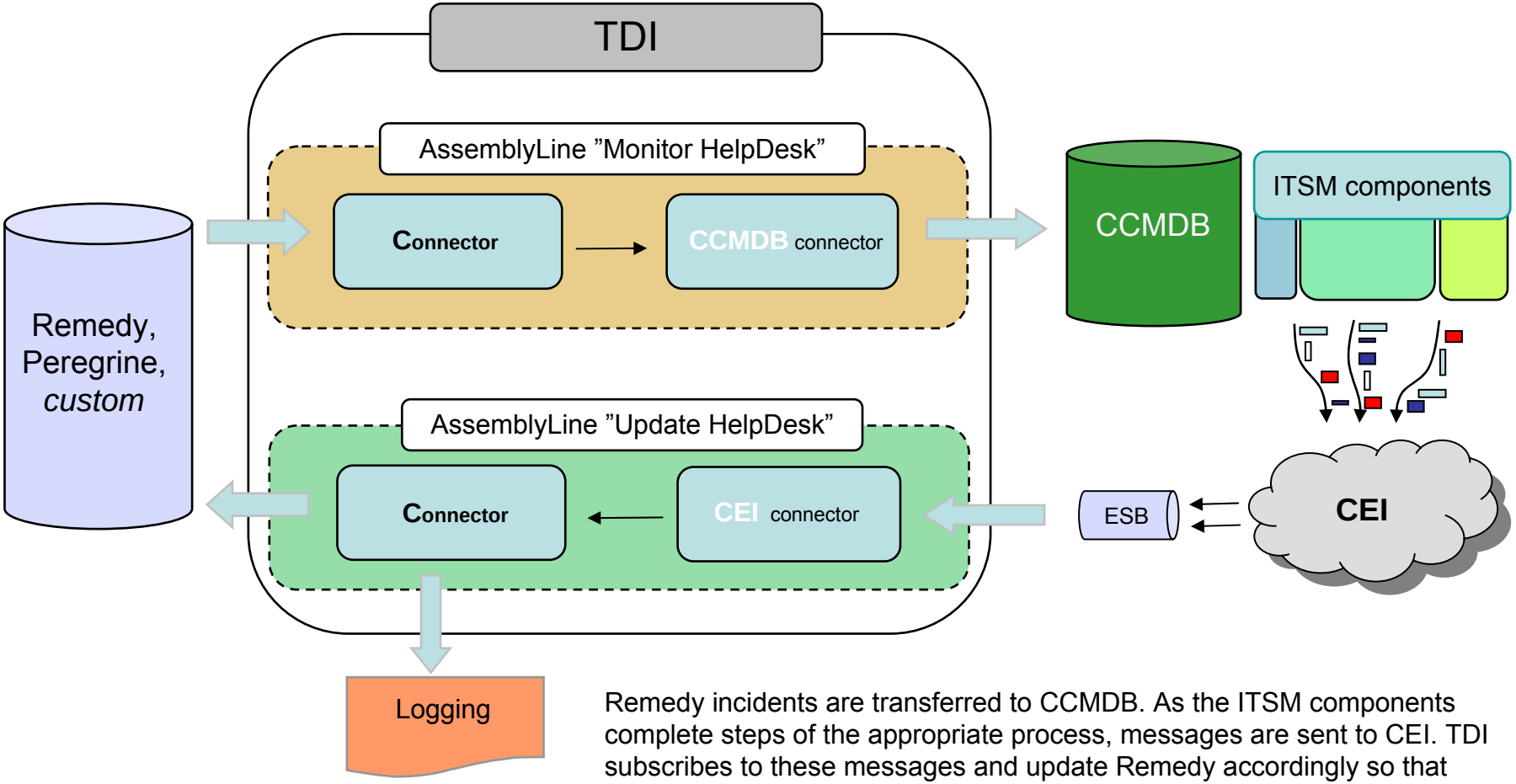


IBM Service Desk integrated with monitoring



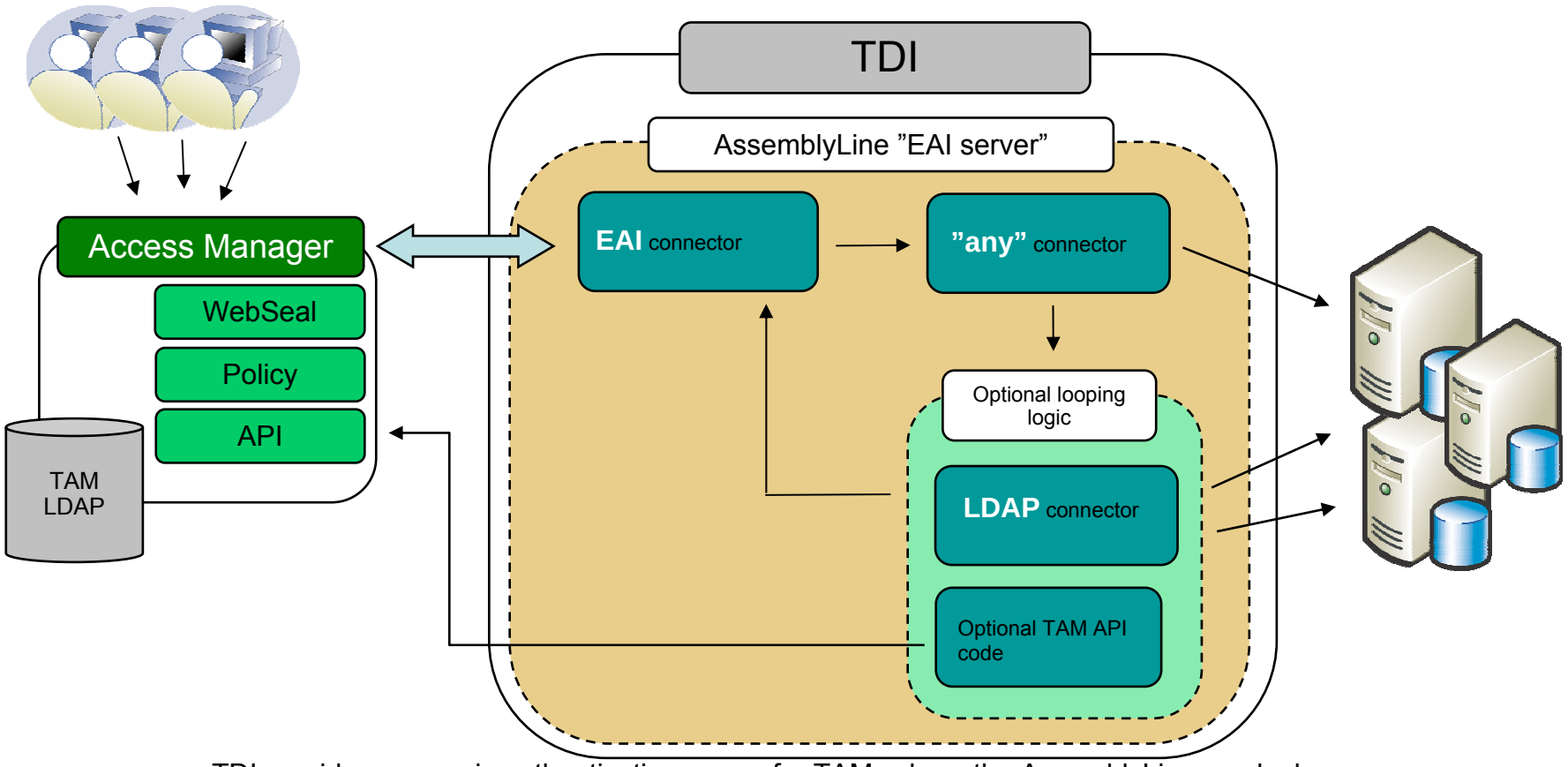
Events (EIF) are augmented with relevant data and produces a ticket in TSRM. When tickets are closed, a corresponding EIF message is sent back that closes the incident

3rd Party Help Desk integrated with CCMDB



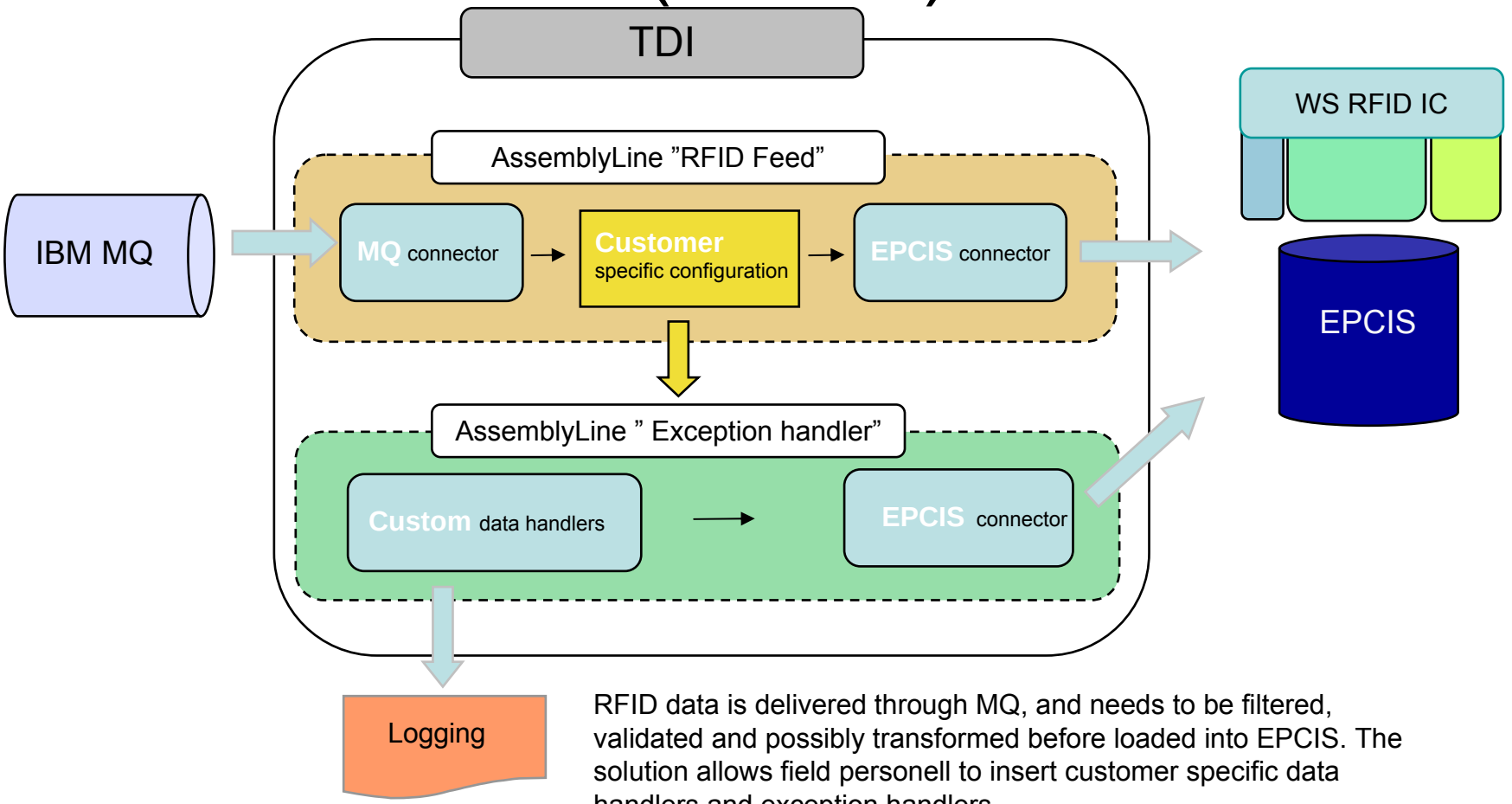
Remedy incidents are transferred to CCMDB. As the ITSM components complete steps of the appropriate process, messages are sent to CEI. TDI subscribes to these messages and update Remedy accordingly so that users can follow the progress of their tickets.

EAI authentication service for TAM



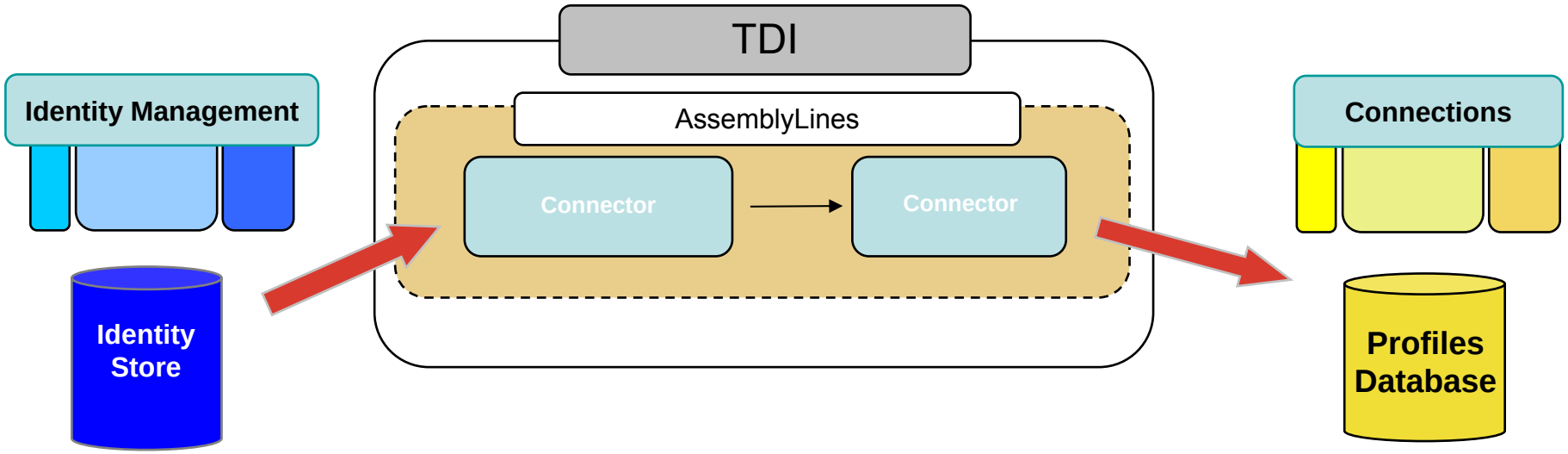
TDI provides a generic authentication server for TAM, where the AssemblyLine can lookup multiple sources, as well as format and transform data. The AssemblyLine above does not illustrate any specific scenario, but illustrates who the integration with the TAM EAI (External Authentication Interface) service works.

Websphere RFID Information Center (EPCIS)



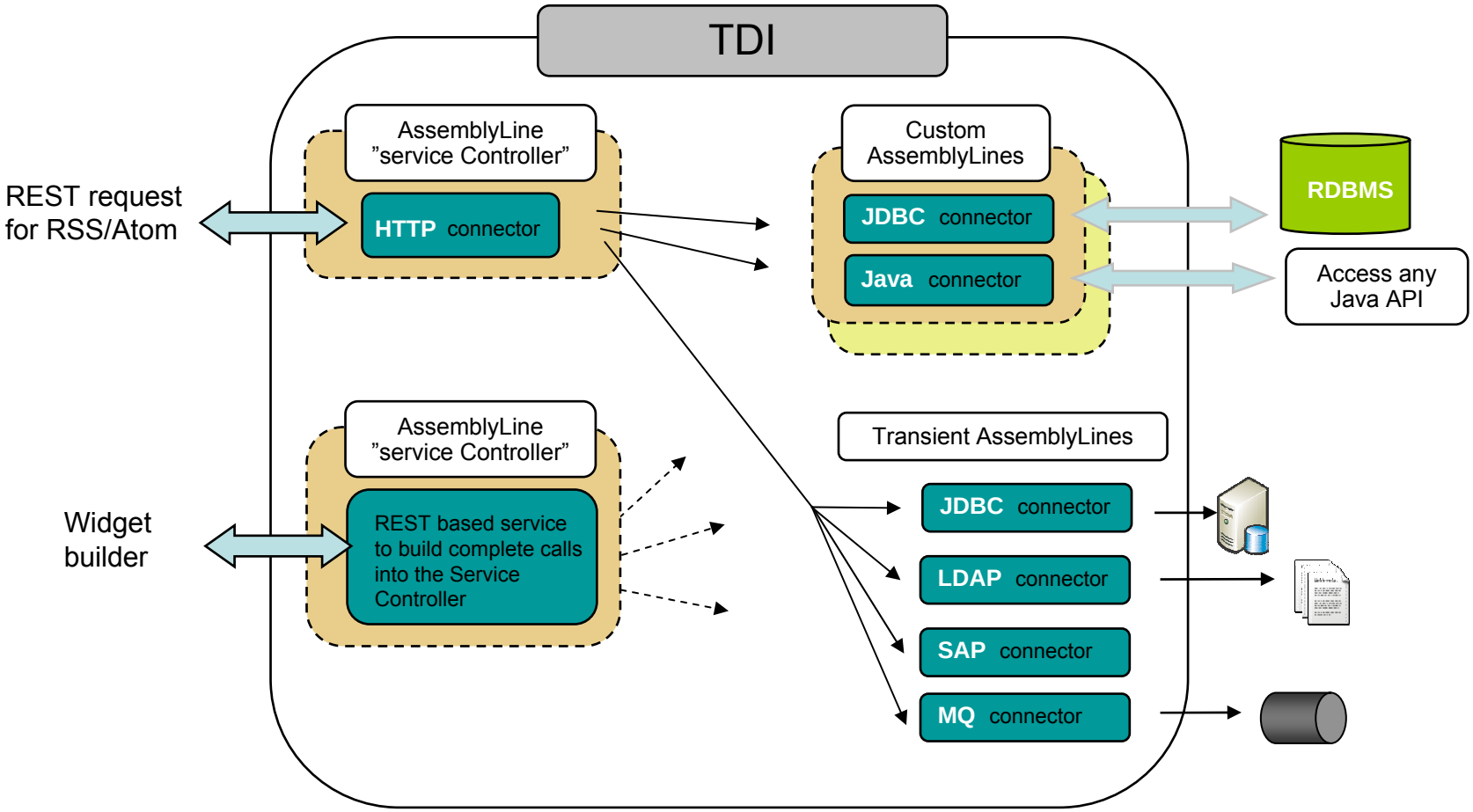
RFID data is delivered through MQ, and needs to be filtered, validated and possibly transformed before loaded into EPCIS. The solution allows field personell to insert customer specific data handlers and exception handlers.

Lotus Connections

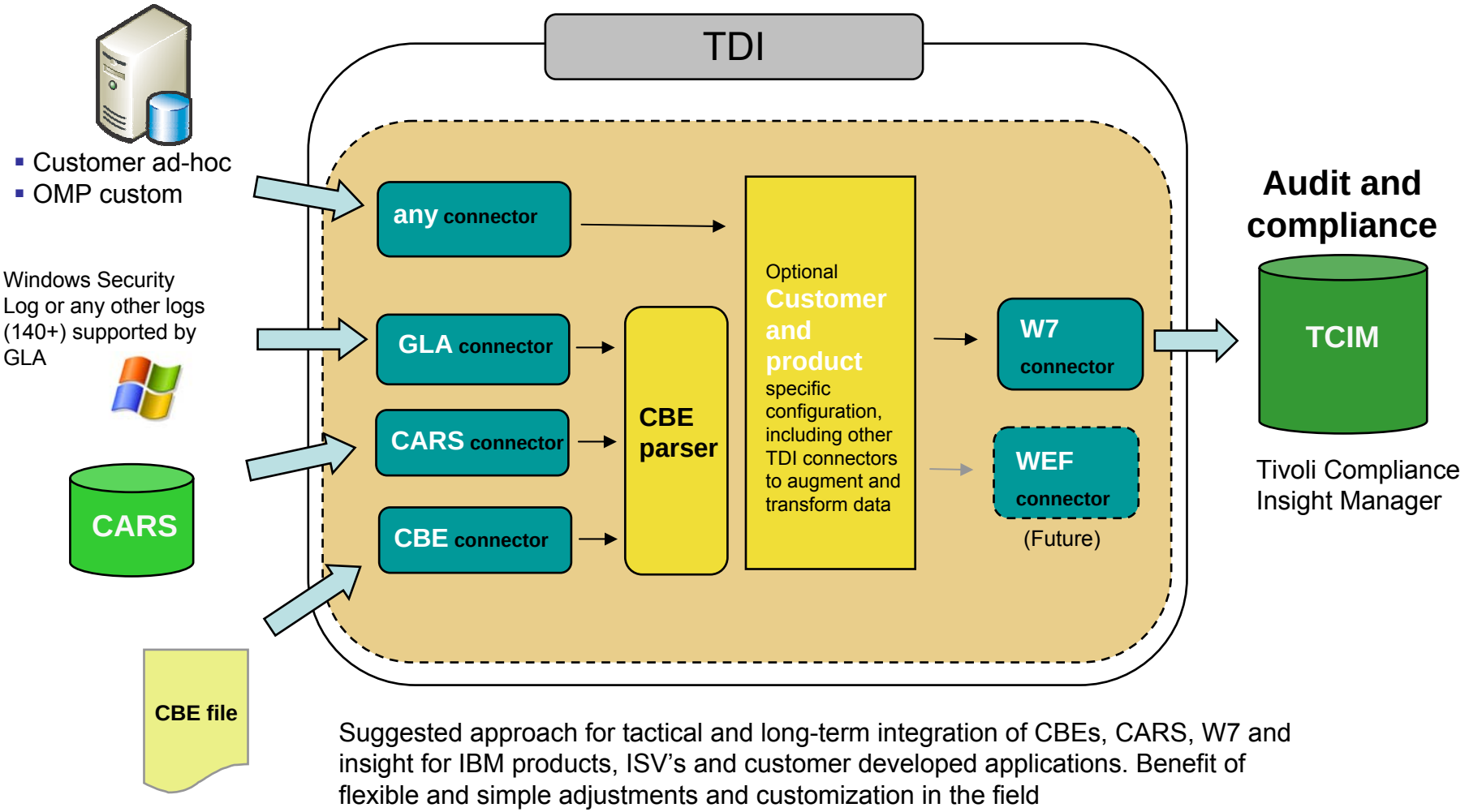


- Initial deployment and provisioning
- Ongoing synchronization

IBM Mashup Center (Lotus Mashups)



TCIM (Insight) - Audit integration



Suggested approach for tactical and long-term integration of CBEs, CARS, W7 and insight for IBM products, ISV's and customer developed applications. Benefit of flexible and simple adjustments and customization in the field

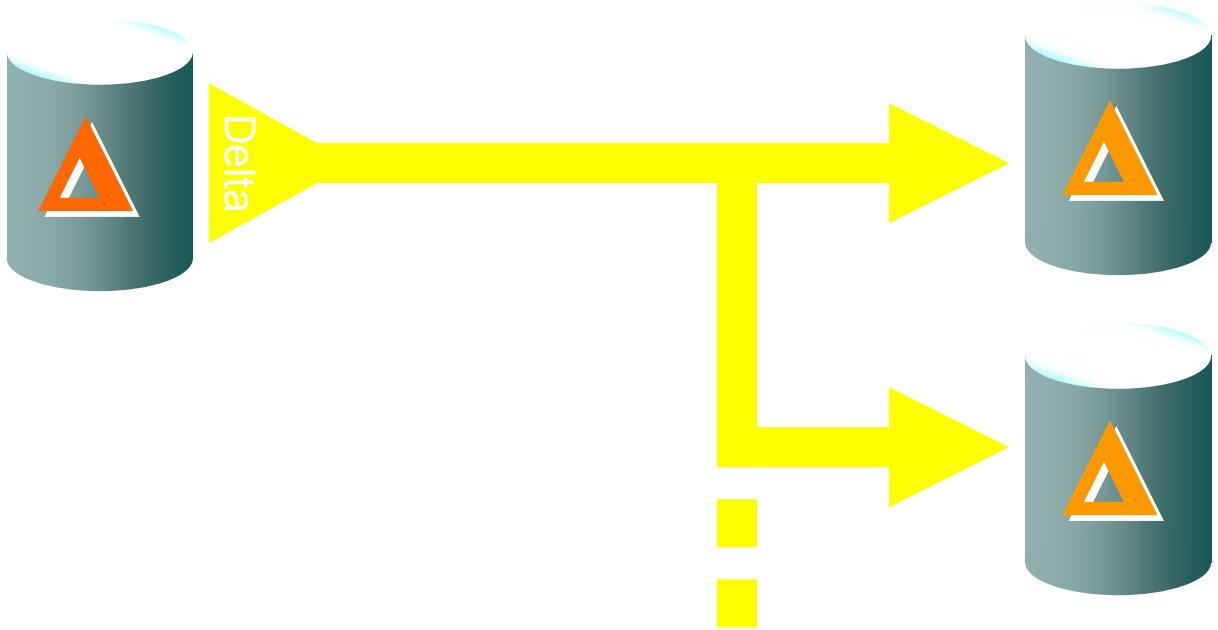
TDI is part of several IBM product offerings

- TSRM (IBM Service Desk)
- IBM CCMDB (Change Configuration & Management DB)
- Lotus Connections
- IBM Mashup Center (incl. Lotus Mashup)
- WebSphere RFID Information Center
- Tivoli Identity Manager
- Tivoli Access Manager for eBusiness
- Lotus Domino 8
- Tivoli Directory Server
- Federated Identity Manager

...

Data Synchronization

- Two distinct activites:
  - Discovering changes in one system
  - Applying these changes to one or more targets.



Delta Detection

TDI provides three mechanisms for Delta Detection:

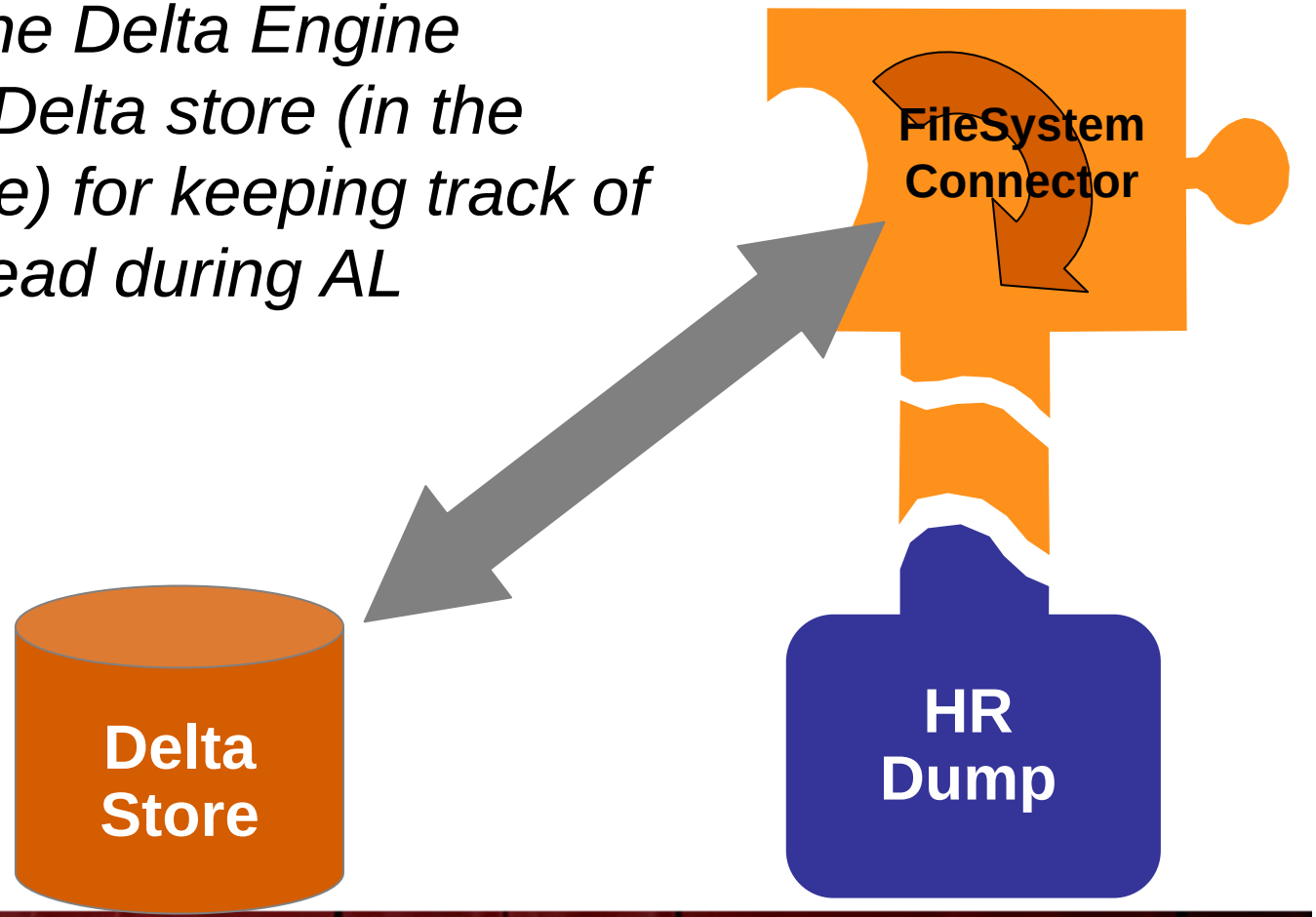
- *Change Detection Connectors*
- *LDIF Parser* *(specifically, incremental LDIF)*
- *Delta Engine*

Change Detection Connectors (CDCs)

- *TDS Changelog*
- *Netscape/iPlanet Changelog*
- *MS Active Directory Changelog*
- *Domino Change Detection*
- *RDBMS Changelog*
- *Exchange Changelog (NT4 support)*
- *Password Sync plugins for Active Directory, TDS, Sun/iPlanet, Domino HTTP, UNIX/Linux and Z/OS (mainframe)*

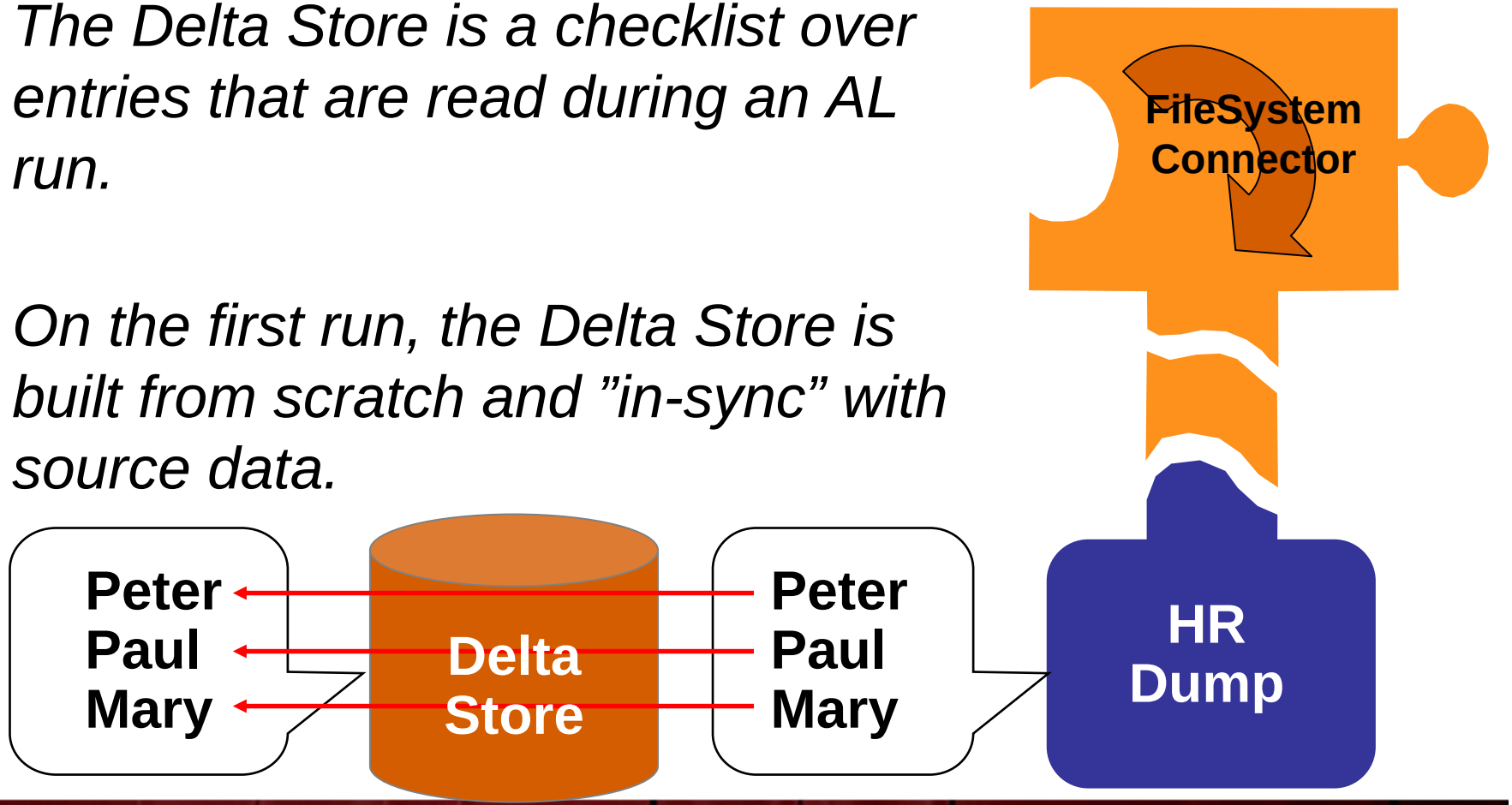
Delta Engine (Iterator mode)

- *If enabled, the Delta Engine maintains a Delta store (in the System Store) for keeping track of each entry read during AL execution.*



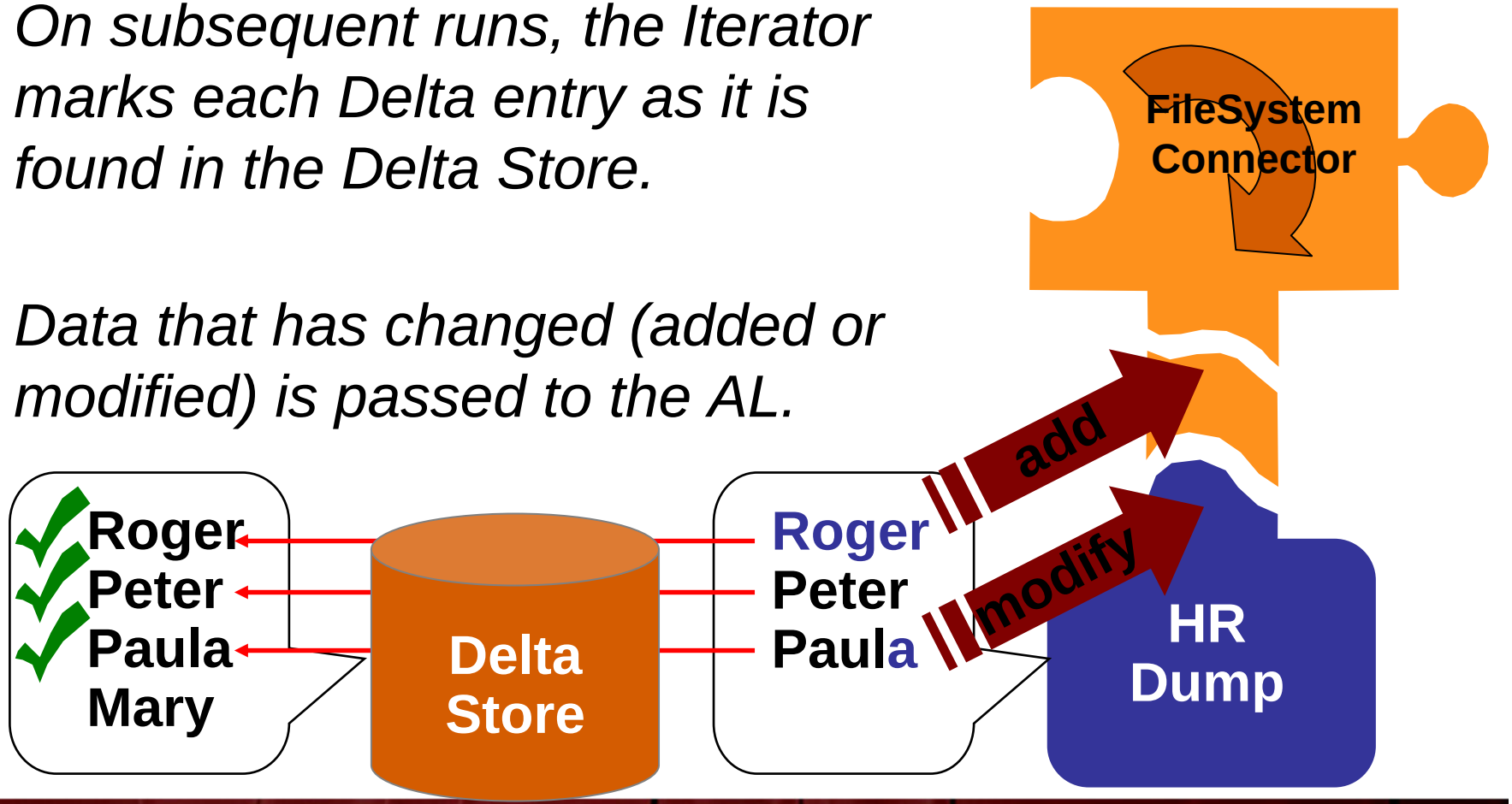
Delta Engine (first run)

- *The Delta Store is a checklist over entries that are read during an AL run.*
- *On the first run, the Delta Store is built from scratch and ”in-sync” with source data.*



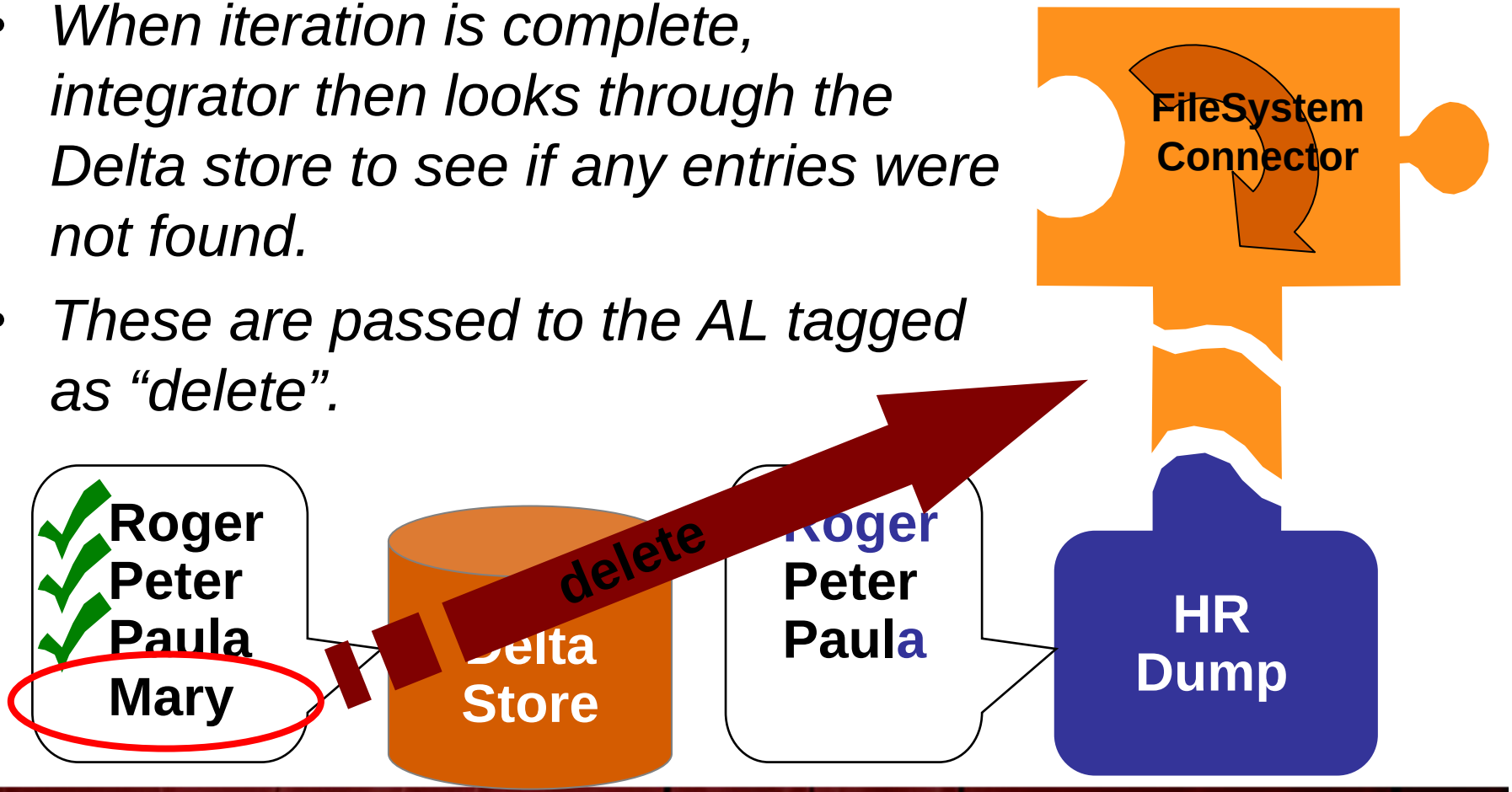
Delta Engine (subsequent runs)

- *On subsequent runs, the Iterator marks each Delta entry as it is found in the Delta Store.*
- *Data that has changed (added or modified) is passed to the AL.*

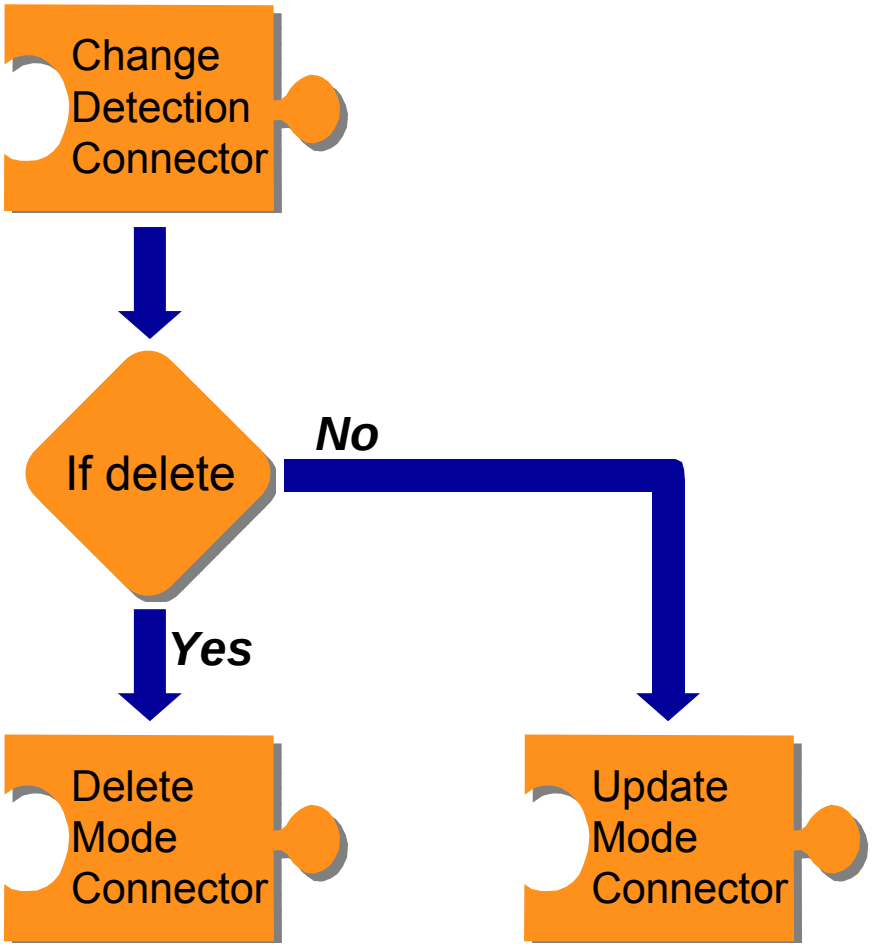


Delta Engine (subsequent runs)

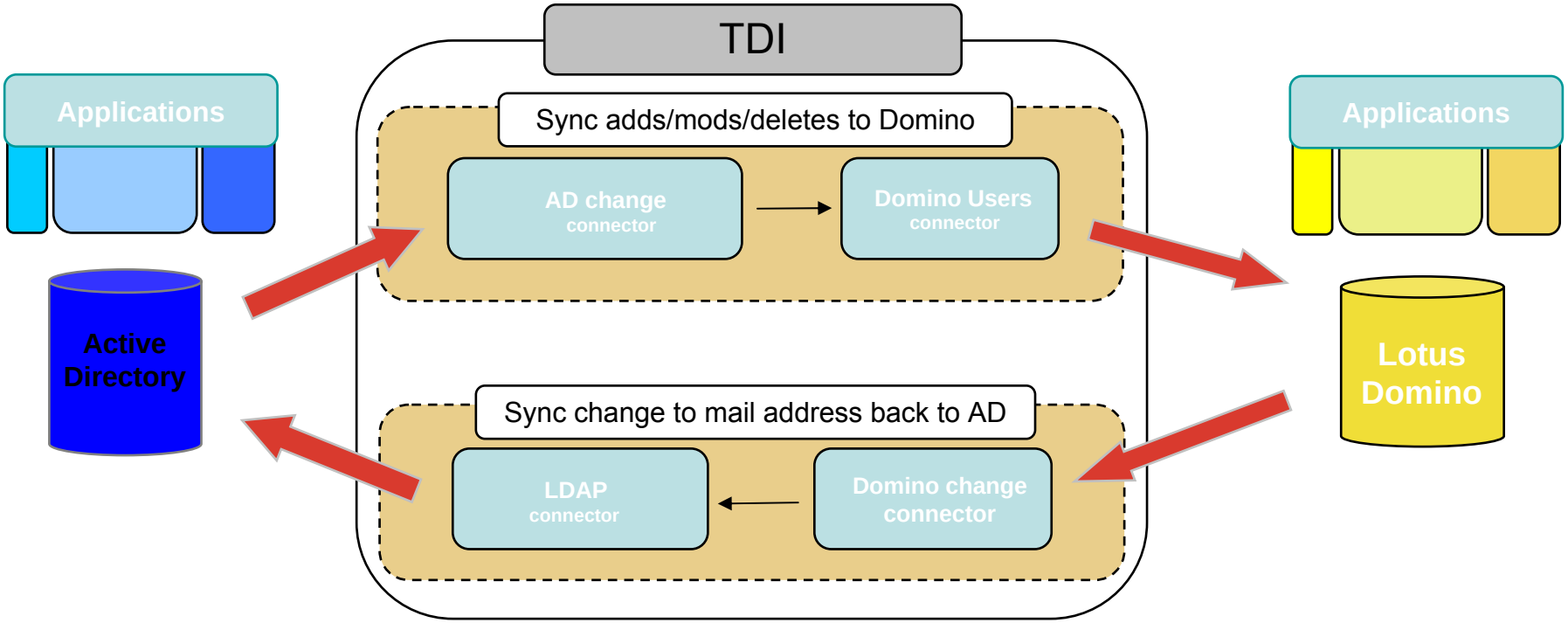
- *When iteration is complete, integrator then looks through the Delta store to see if any entries were not found.*
- *These are passed to the AL tagged as “delete”.*



Synchronization AL logic



Bi-directional AD - Domino Sync



In this scenario, users are managed in AD and need to be synchronized with Domino

1. All data is pushed into Domino when new Users are added in AD.
2. E-mail is ”owned” by Domino, so changes in AD will not overwrite the ”Internet Address” in Domino.
3. Changes to the internet mail address in Domino are sync’ed to AD.
4. Any of the above logic can be easily and quickly modified and extended.

TDI Resources (pass these one to colleagues, partners and users)

- Forums
  - External: news://news.software.ibm.com/ibm.software.network.directory-integrator
  - IBM Internal: news://forums.ibm.com/forums.software.directory
- Online docs, including the Getting Started Guide
  http://publib.boulder.ibm.com/infocenter/tivihelp/v2r1/index.jsp?topic=/com.ibm.IBMDI.doc_6.1.1/welcome.htm
- TDI community assets
  - HowTo's guides, like these on data synchronization features and error handling:
    http://www.tdi-users.org/twiki/pub/Integrator/HowTo/HowTo_SyncData_6.1.1070523.pdf
    http://www.tdi-users.org/twiki/pub/Integrator/HowTo/HowTo_HandleErrors.pdf
  - TDI Flow Diagrams:
    http://publib.boulder.ibm.com/infocenter/tivihelp/v2r1/topic/com.ibm.IBMDI.doc_6.1.1/TDI_6.1_FlowDiagrams.pdf
- Video Tutorials
  - External link for partners and customers
    http://**www.tdi-users.org**
  - IBM internal
    https://**w3.webahead.ibm.com/w3ki/display/TDI**