Previews of TDWI course books are provided as an opportunity to see the quality of our material and help you to select the courses that best fit your needs. The previews can not be printed.

TDWI strives to provide course books that are content-rich and that serve as useful reference documents after a class has ended.

This preview shows selected pages that are representative of the entire course book. The pages shown are not consecutive. The page numbers as they appear in the actual course material are shown at the bottom of each page. All table-of-contents pages are included to illustrate all of the topics covered by a course.

TDWI Data Integration Techniques

ETL and Alternatives for Data Consolidation

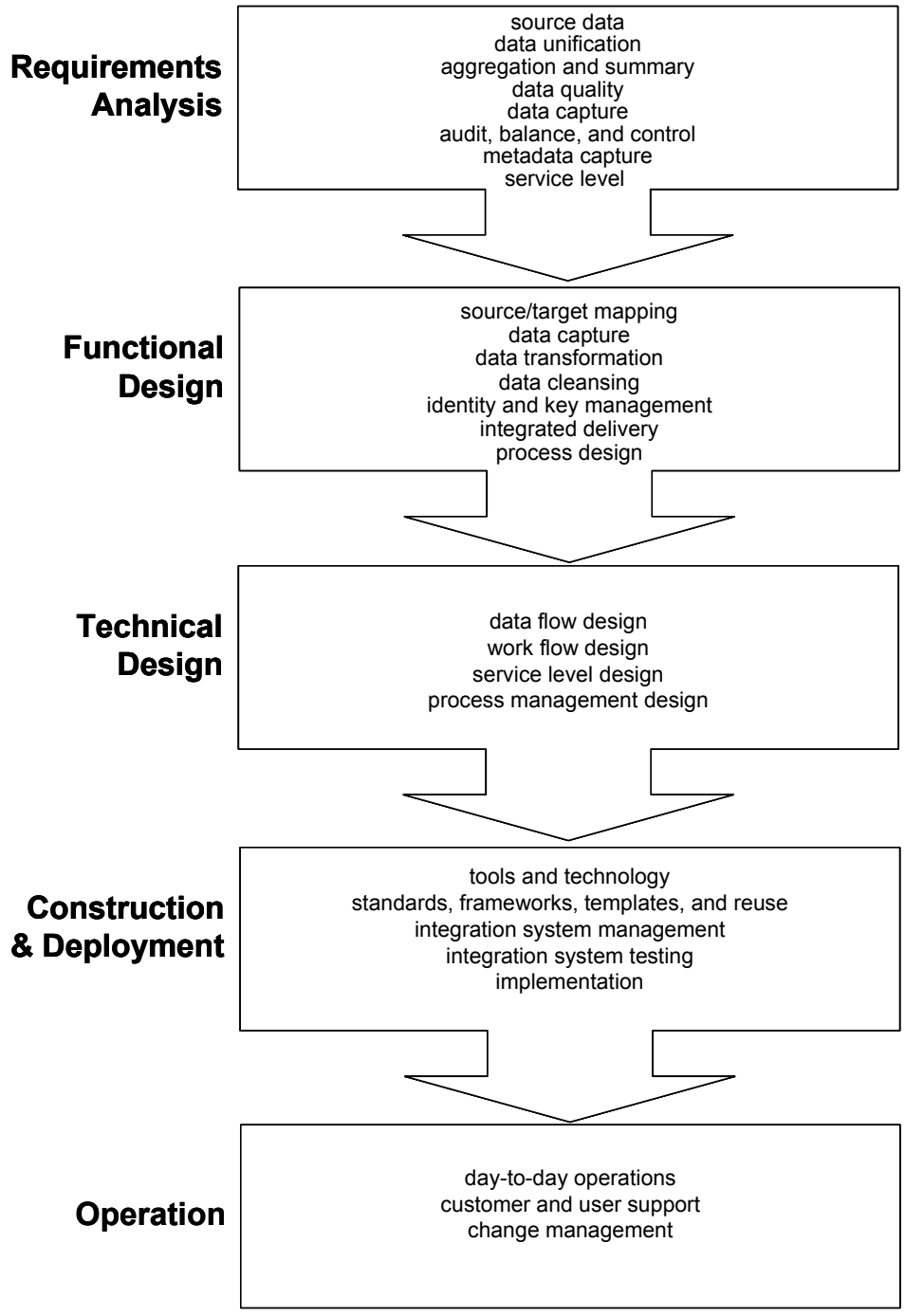
Module 1

Data Integration Concepts

This page intentionally left blank.

Data Integration Projects

Project Activities



Data Integration Projects

Project Activities

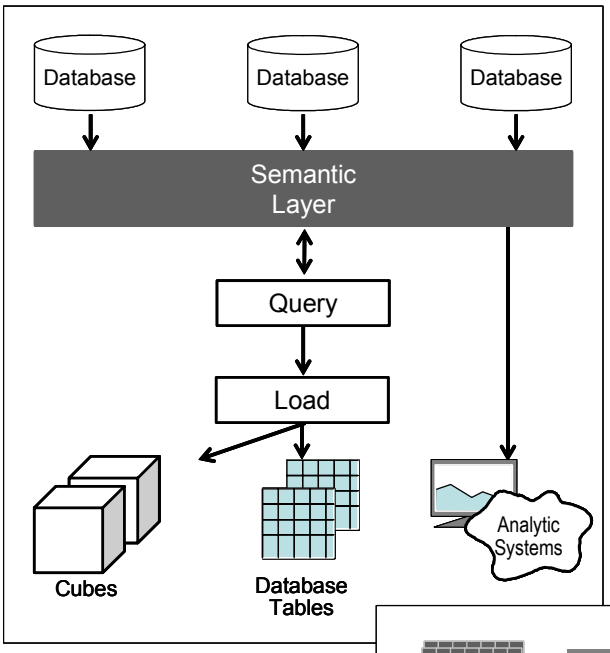
STEPS TO INTEGRATED DATA

As with any information system or project, data integration has a lifecycle that follows a progression from requirements to implementation and operation. Each stage of the lifecycle performs a set of activities to produce specific results.

The diagram on the facing page illustrates a six-phase lifecycle for data integration projects. For each phase the major topics related to activities and results are shown. The rest of this course takes and in-depth look at all of these topics in the sequence illustrated here.

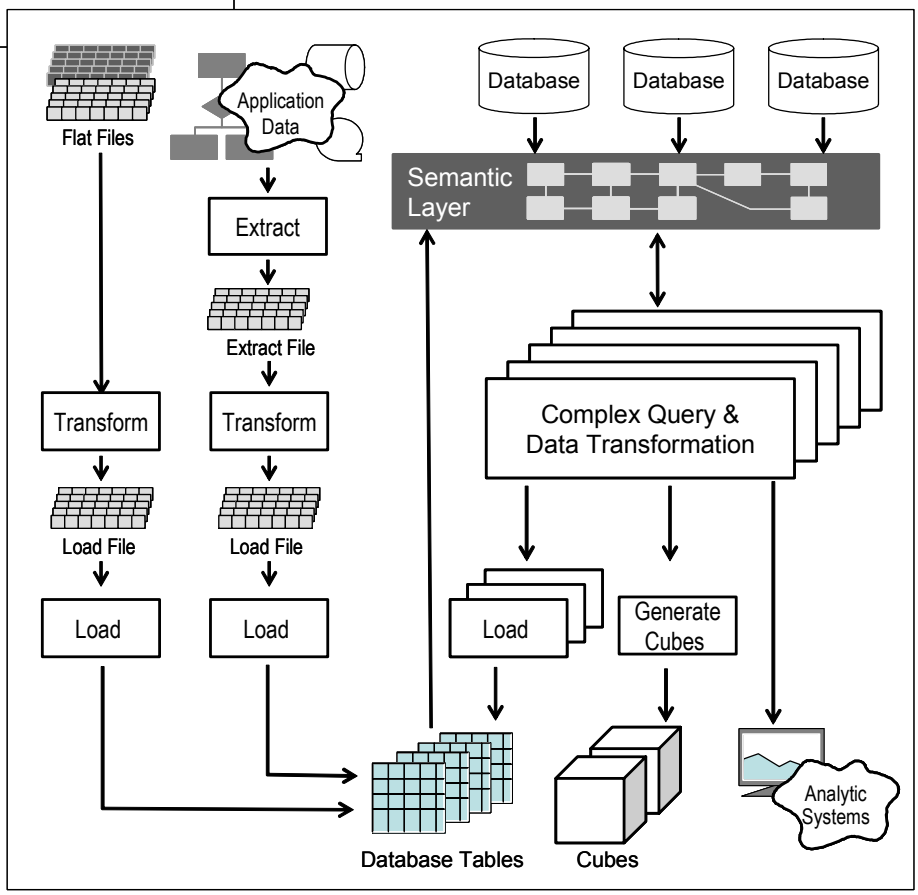
Data Integration Technologies

Enterprise Information Integration (EII)



← THE SIMPLE VIEW OF EII

THE REALITY OF EII →



Data Integration Technologies

Enterprise Information Integration (EII)

ON-DEMAND DATA INTEGRATION

Enterprise information integration (EII) is sometimes described as virtual ETL. Using a semantic layer to support data abstraction and to provide business context, EII performs on-demand data integration. Complex queries that include limited data transformation capabilities are the heart of EII. Data is accessed, transformed, and delivered (to analytic applications, cube generators, or database load processes) in one logical operation. One execution of an EII operation works with a much smaller set of data than a single execution of ETL.

WHEN TO USE EII

EII is most effective to meet demands for small amounts of real-time data, with data warehousing, business analytics, and MDM as the most practical applications. For data warehousing applications EII is effective as a complement to ETL, using ETL where latency of data is acceptable and EII where real-time data is needed.

Philip Russom, a research analyst at TDWI, describes EII as something like a freeway bypass. It can "help get data to the data warehouse, and bypass the data warehouse for quick refresh data if needed. Ideally, the rules for the EII refresh are the same as the rules for the ETL job that loads the daily grain data, and the metadata is shared so it's all traceable from within the same environment."

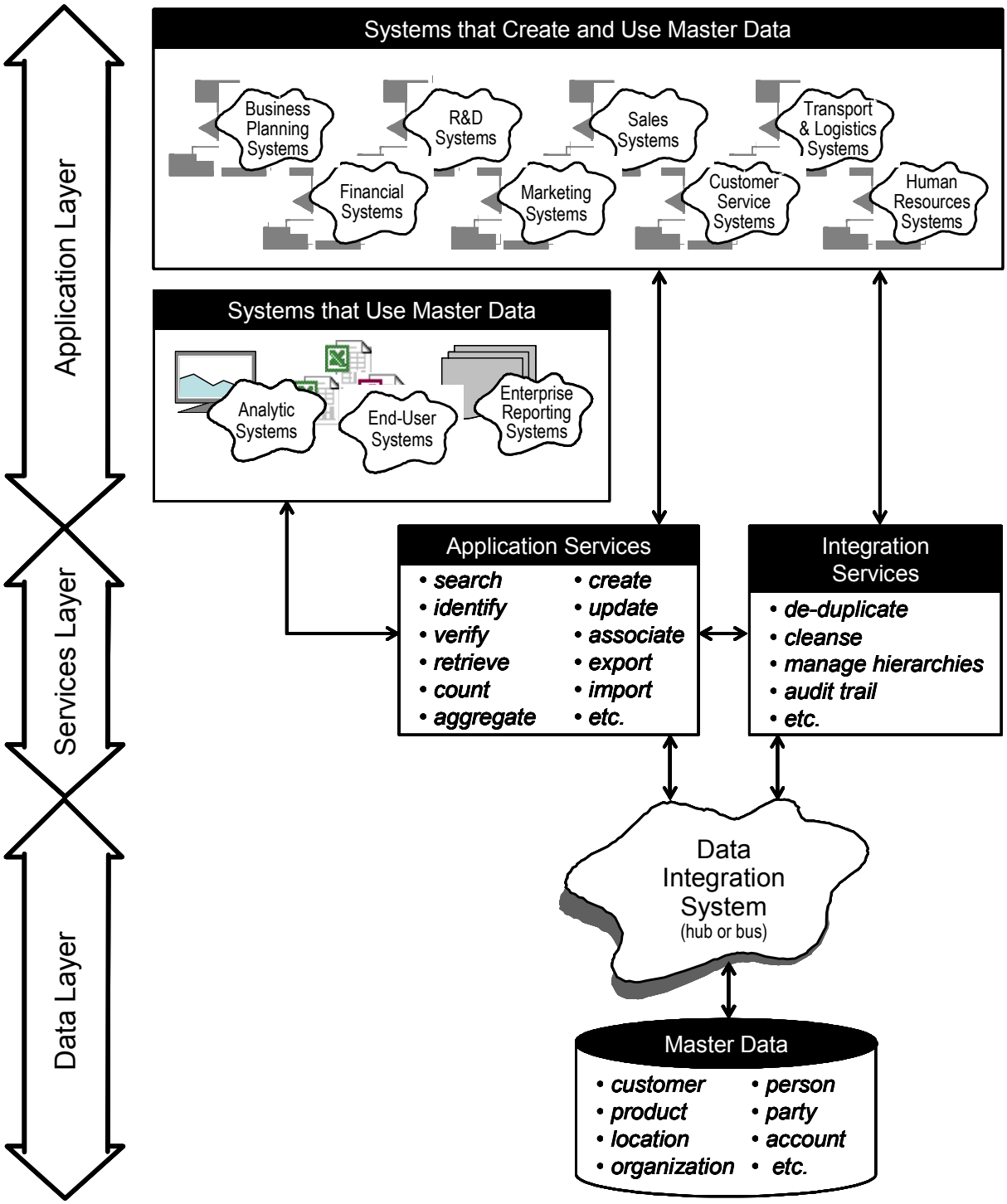
EII is also emerging as one answer to the challenges of integrating text and documents into the BI environment. These types of unstructured data add depth to the information resource but require little transformation. EII can be used to connect them with BI while avoiding the cost of ETL and redundant storage to move them into a data warehouse.

EII TECHNOLOGY

Many EII-like capabilities can be implemented using database features such as materialized views (Oracle), materialized query tables (DB2), linked servers (SQLServer), etc. More robust semantic layers and more advanced EII capabilities are supported in EII-specific vendor tools such as Composite, Metamatrix, and Ipedo.

Data Integration Technologies

Master Data Management (MDM) and More



Data Integration Technologies

Master Data Management (MDM) and More

INTEGRATING DATA AT THE CORE OF THE ENTERPRISE

MDM encompasses the disciplines, standards, and technologies that integrate enterprise reference data to provide a consistent view for all departments and applications. Most organizations have core reference data such as data about customers, products, locations, etc. distributed throughout many applications that manage their databases independently. When core reference data can't be reconciled across applications and databases enterprise reporting and compliance monitoring are especially challenging. MDM establishes a system-of-record for enterprise-critical reference data.

IMPLEMENTING MDM

MDM combines data, services, and integration to address the need for a system-of-record. The services layer provides both integration services to unify and cleanse data an application services to access, create, and modify data. Integration services require a data integration system which may use either hub-and-spoke or bus architecture similar to those that are common in data warehousing. Unlike data warehousing, MDM is bidirectional integration – data may be integrated at both inbound and outbound points of service.

MDM is neither a replacement for nor a component of a data warehouse. It is a unique integration solution that is compatible with warehousing but with distinctly different business purpose. The data warehouse may, in fact be part of the problem that MDM solves; In many organizations the warehouse becomes just one of many inconsistent sources where master data is stored.

MDM VARIATIONS

Customer data integration (CDI) is a variation of MDM that focuses specifically on creating a system-of-record for customer data. Customer data is perhaps the most widespread of reference data integration problems. By focusing only on customer, data management discipline becomes more rigorous and integration tools are able to provide customer-specific features and functions.

Product information management (PIM) is a variation similar to CDI but focused exclusively on enterprise integration of product data.

MDM TECHNOLOGY

MDM products and services have been released by many established data integration and cleansing software vendors – IBM and Informatica, for example. In addition new vendors have emerged with MDM-specific products. Expect this relatively new technology to continue to evolve.

Module 2

Requirements Analysis for Data Integration

This page intentionally left blank.

Source Data Requirements

Source Data Analysis and Profiling

	data field/column	description (what fact?)	entity (fact about?)	identifier?	comments
E-Max Employee Table	employee_id	unique id number for each employee	EMPLOYEE	yes	
	employee_name	legal name as shown on the payroll	EMPLOYEE		
	date_of_birth	date that the employee was born	EMPLOYEE		
	sex	male or female?	EMPLOYEE		
	address_line1	1st line of mailing address	EMPLOYEE		
	address_line2	2nd line of mailing address	EMPLOYEE		optional
	city	city of mailing address	EMPLOYEE		
	state	state of mailing address	EMPLOYEE		
	zip_code	zip code of mailing address	EMPLOYEE		
	ethinc_origin_code	code describing employee's ethnic background	EMPLOYEE		
	federal_tax_marital_status	marital status claimed on IRS form W4	EMPLOYEE		
	federal_tax_number_of_exemptions	number of exemptions claimed on IRS form W4	EMPLOYEE		
	state_tax_marital_status	marital status claimed for state taxes	EMPLOYEE		optional - depends on state of residence
	state_tax_number_of_exemptions	number of exemptions claimed for state taxes	EMPLOYEE		optional - depends on state of residence
	hire_date	date of first day of employment	EMPLOYEE		
	separation_date	date of last day of employment	EMPLOYEE		only for retired or separated
	employment_status_code	active, inactive, retired or separated	EMPLOYEE		
	employment_status_date	date associated with status code	EMPLOYEE		
	SSN	social security number	EMPLOYEE		sed as the employee ID

	data field/column	description	entity		
PlayNation Employee Table	social_security_number	employee's social security number	EM		
	first_name	employee's given name			
	last_name	employee's surname			
	middle_initial	initial of employee's midd			al
	birthdate	date employee wa			
	gender	male or fem			
	mailing_address	stree			
	city		YEE		
	state		MPLOYEE		
	zip_code		EMPLOYEE		
	home_phone_n		EMPLOYEE		
	work_phor		EMPLOYEE		
	emergency_	ncy contact person	EMPLOYEE		optional
	emergency_c	emergency contact person	EMPLOYEE		optional
	tax_status_fed	ital status claimed on W4 form	EMPLOYEE		
	tax_exemptions_	exemptions claimed on w4 from	EMPLOYEE		
	tax_status_state	tax status claimed for state taxes	EMPLOYEE		optional depending on state
	tax_exemptions_st	exemptions claimed for state taxes	EMPLOYEE		optional depending on state
	employment_date	date of first day of employment	EMPLOYEE		
	annual_salary	annual salary to nearest $1000	EMPLOYEE		derived for non-salaried employees
	health_insurance_enrolled_indicator	enrolled in health plan (yes/no)	BENEFITS		
	spouse_health_indicator	spouse health coverage (yes/no)	BENEFITS		
	dependent_health_indicator	dependent health coverage	BENEFITS		number of insured dependents
	ESP_deduction_amount	savings plan deduction amount	BENEFITS		
	profit_sharing_eligiblility_boolean	eligible for profit sharing plan (yes/no)	BENEFITS		
	comments	freeform remarks about employee	EMPLOYEE		
	local_field_1	locally defined field			use determined by each division
	local_field_2	locally defined field			use determined by each division

know the data definitions and the business rules for every field & column in every file & table that will be used as a source for integrated data

Source Data Requirements

Source Data Analysis and Profiling

EXTRACTING THE DATA STRUCTURE

Module One of this course described the many challenges of source data – poor design, undocumented, misused, deteriorating structure and quality, incomplete, inconsistent, etc. – and emphasized the need to fully understand the source data. To use any data source effectively you need to know the data definitions and the business rules for every field and column in every file and table that will be use as a source of integrate data. This essential task (some call it "data archaeology") begins by understanding the structure of the data. A source data element matrix such as the illustration on the facing page is an effective way to begin understanding source data structures.

DOCUMENTING THE DATA ELEMENTS

Gathering data definitions is more tedious than difficult. First get a complete list of the tables/files and columns/fields to be used as source data. Then start with the assumption that every column in every table (or every field in every file) is a fact about something. Then answer three questions about each column/field:

1. What fact does the field or column contain?
2. What thing is it a fact about? (What does the fact describe?)
3. Does the field or column uniquely identify the thing that it describes?

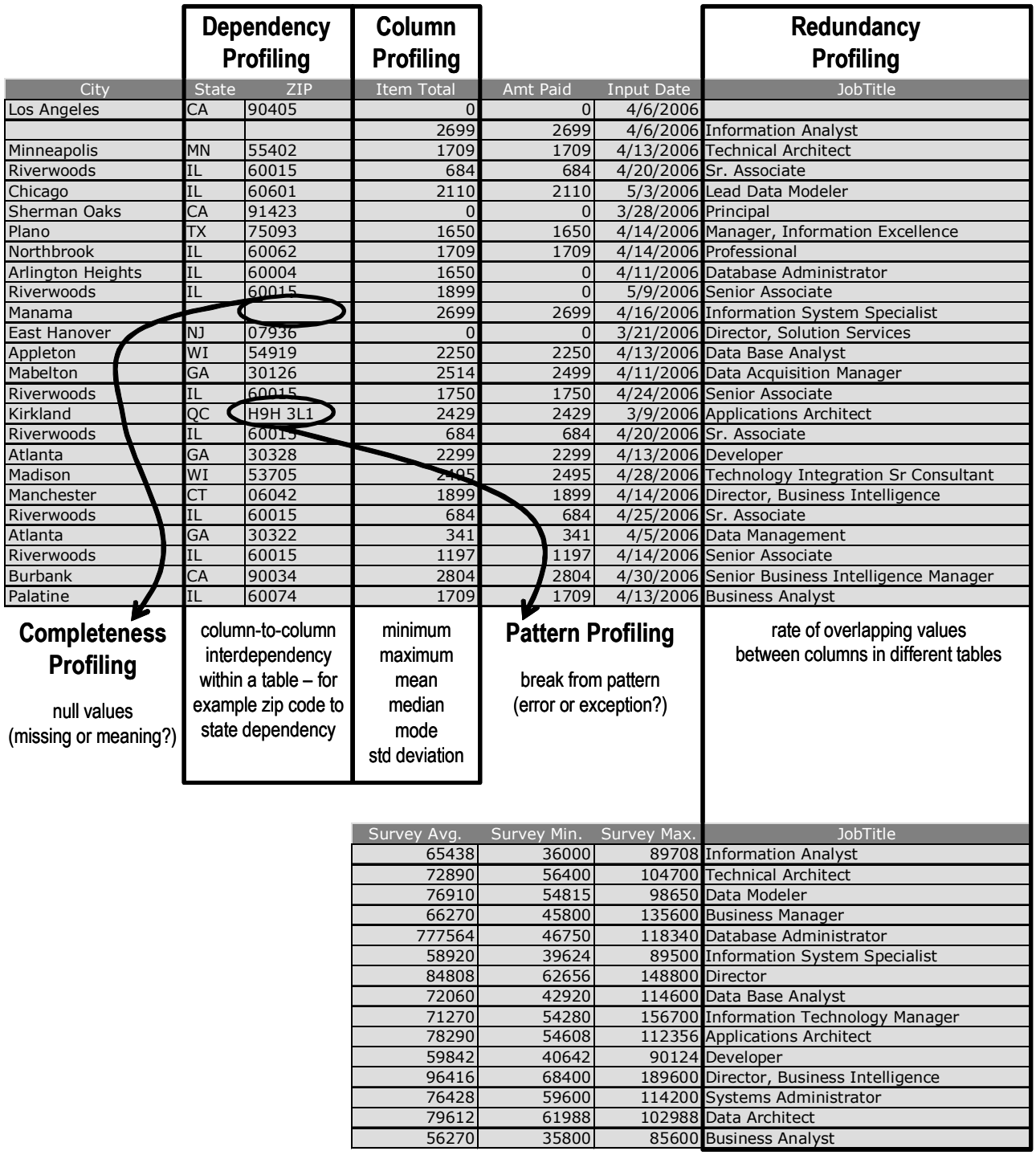
You will, of course, encounter some difficulty getting clear answers to these questions. Sometimes it takes a lot of digging and a bit of discussion to resolve the questions. Tenacity pays off, however. If you don't know what the data means, how can it safely be integrated with other data or used to make decisions?

FINDING THE BUSINESS RULES

Once definitions are known, the next step in understanding source data is to examine the contents – the specific values – contained in the data and begin to understand what governs those values. Almost every data element has some rules that control what values may be placed into it. Although commonly called "edit rules" and thought of as computer stuff, every edit rule has it's origin in a business rule. Collecting business rules now helps to determine how data can be integrated, and to define data transformation rules later.

Source Data Requirements

Source Data Analysis and Profiling



Source Data Requirements

Source Data Analysis and Profiling

UNDERSTANDING THE DATA CONTENT

Understanding the basic structure and knowing the definitions and business rules is a good start, but it isn't enough to use a data source with confidence. Hidden structure, unexpected uses, and quality issues can be known only by looking at data contents.

DATA PROFILING

The process of systematically looking at data to identify and discover patterns is called *data profiling*. Data profiling examines data to understand its content, structure, and dependencies. Profiling explores data in three ways:

- *Column profiling* examines the values and characteristics of data elements. Results of column profiling are information such as minimum value, maximum value, distribution of values, range of values, gaps and missing values in an apparent range, etc.
- *Dependency profiling* identifies element-level connections in the data. It discovers elements with common domains and values to find hidden keys and relationships.
- *Redundancy profiling* examines data to discover duplication of the same data items.

Profiling techniques include pattern recognition and data classification as described below.

PATTERN RECOGNITION

Pattern recognition discovers hidden patterns inherent in data. It is useful to discover hidden data quality rules, prepare to classify data, and define probability algorithms. Common patterns include:

- distribution of values (most frequent to least frequent values)
- affinity of values (when value^1=x, then value^2 is frequently y)
- disparity of values (when value^1=x then value^2 is seldom y)
- similarities and differences (of spelling, abbreviation, rounding, etc.).

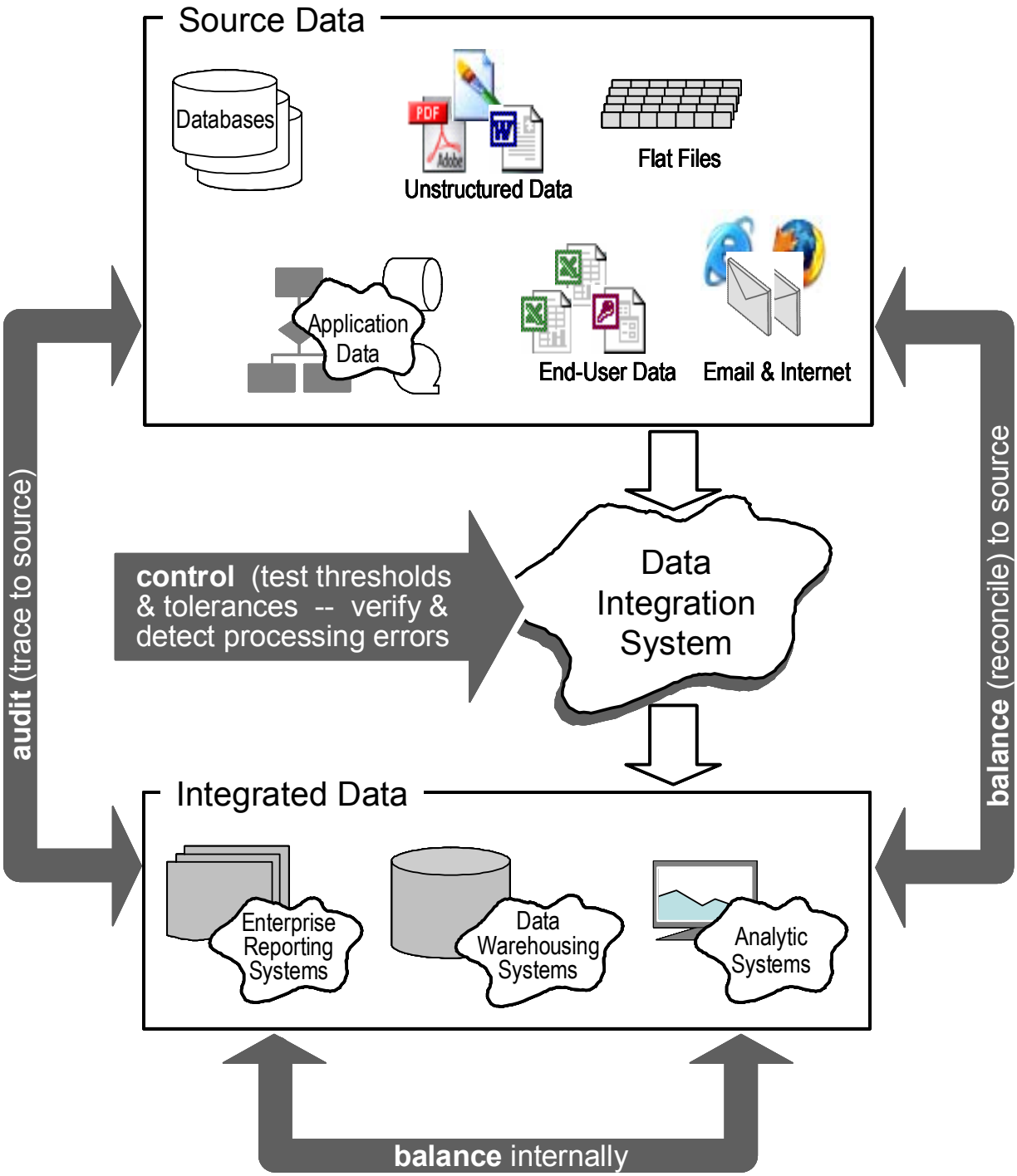
DATA CLASSIFICATION

Classification groups data with common characteristics, and segregates data with unique characteristics. Classification is useful to cluster data with common business properties, common domains, or common quality properties. Common classifications include:

- by business subject
- by domain of values
- by business properties
- by data quality properties
- by business process
- by system process.

Audit, Balance, and Control Requirements

ABC's of Data Integration



Audit, Balance, and Control Requirements

ABC's of Data Integration

TRACEABILITY, RECONCILIATION, AND PROCESS CONTROLS

The ABC's of data integration describe:

- the degree to which integrated data is readily audited (ability to trace back to source data),
- to which data standards or data sources it will balance (and with what tolerance), and
- the level of run-time controls to support early error detection.

Balancing is a common user expectation, even the occasional demand to balance to two different data sources that don't balance between themselves. There will, of course, be differences when any amount of data transformation occurs. The key to balancing is *explainable* differences. And the key to explainable differences is audit capability.

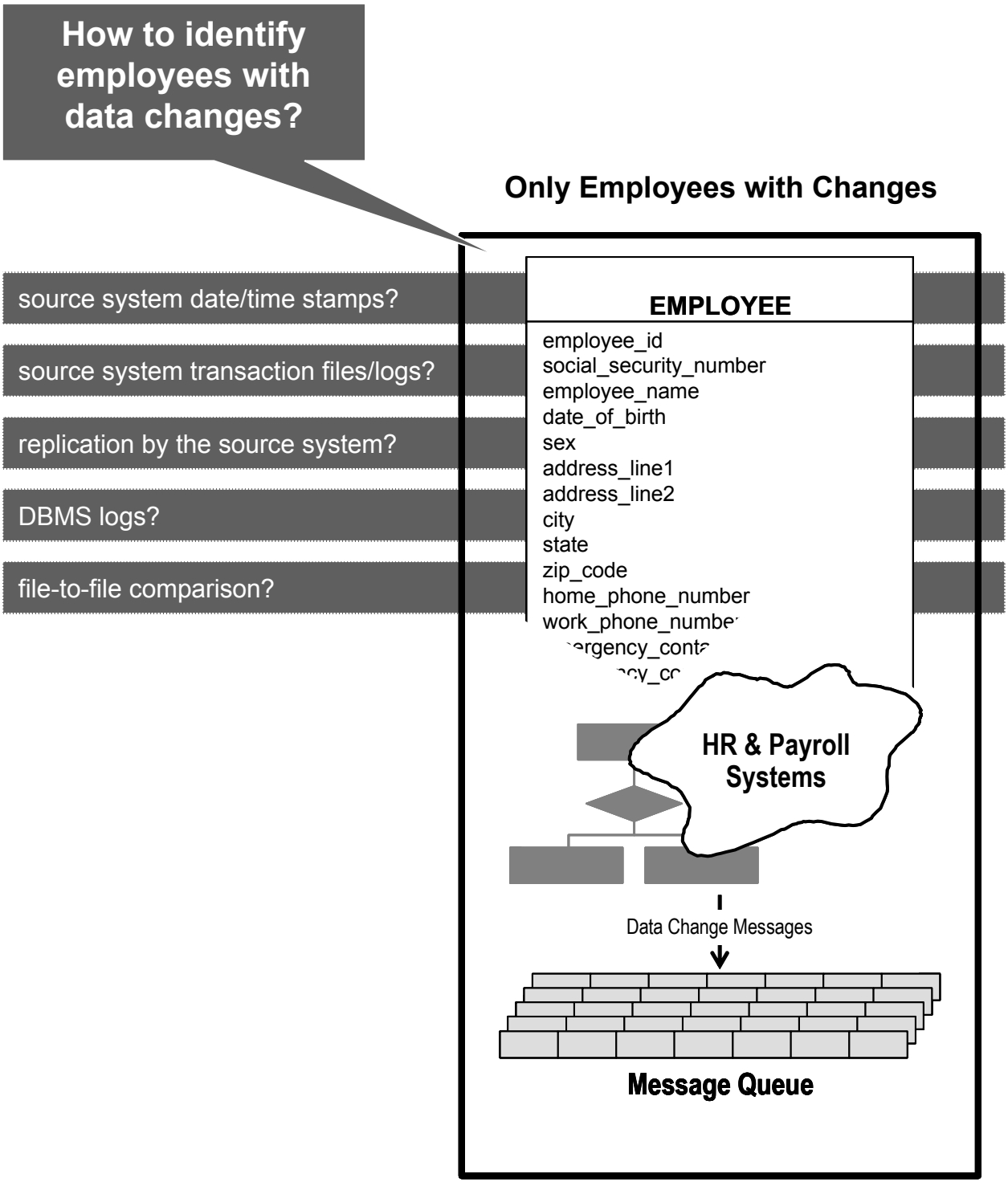
Module 3

Data Integration Functional Design

This page intentionally left blank.

Data Capture Design and Specification

Changed Data Detection



Data Capture Design and Specification

Changed Data Detection

DETECTION RESPONSIBILITY

Detecting data changes is often challenging. The first decision for any source requiring change detection is where the responsibility of detection resides. Is the source system responsible to report all changes to the data integration system? Or is the integration system responsible to identify what has changed by examining the data at the source?

In general, placing responsibility with the integration system is only effective for high-latency warehousing data or periodic data synchronization. Low-latency warehousing, MDM, and near real-time synchronization need to have change detection occur at the source system.

DETECTION BY THE SOURCE SYSTEM

Several techniques are common for change detection at the source. They vary in their ability to meet low-latency data integration needs. Among the techniques commonly applied:

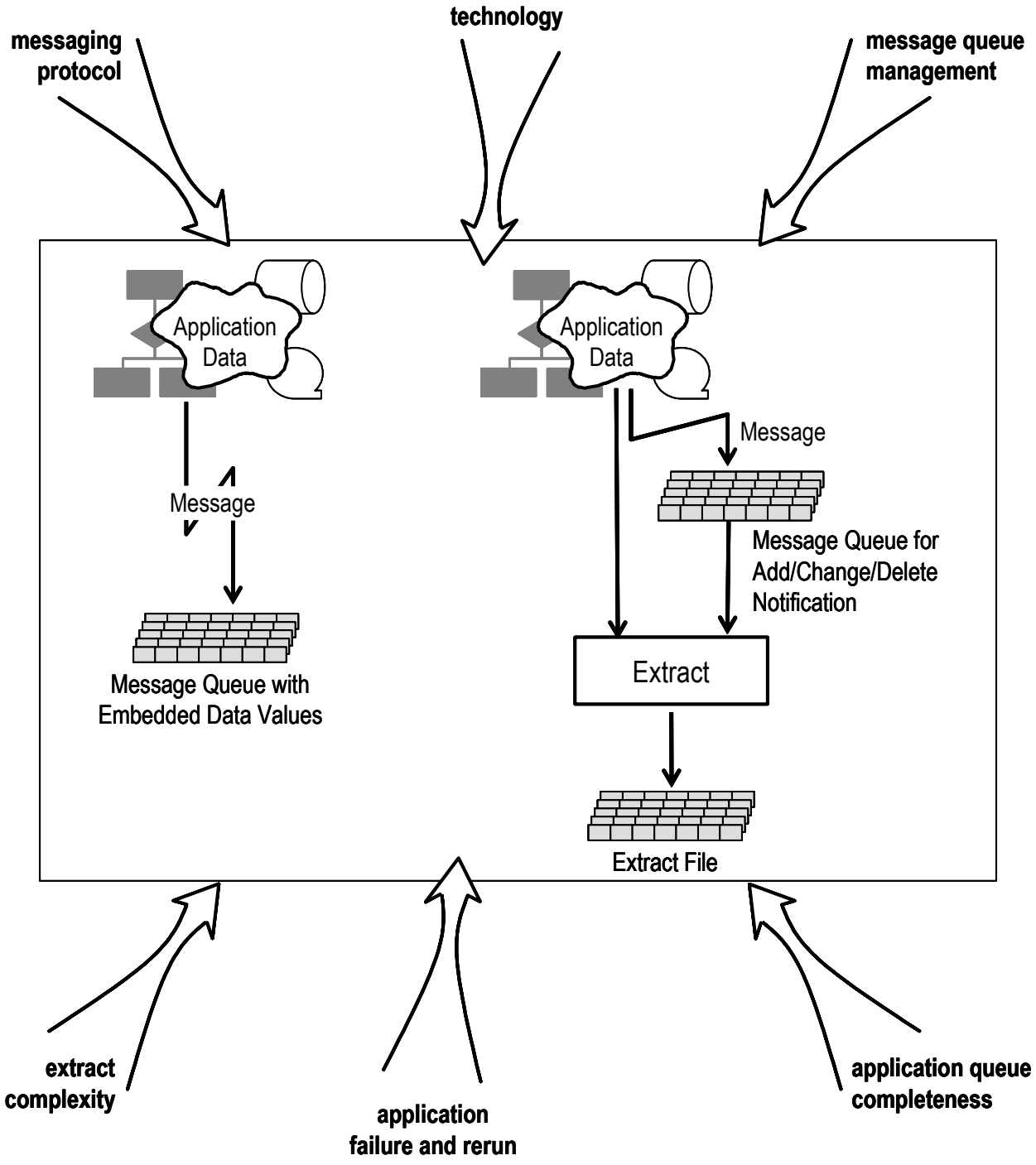
- Use source date/time stamps that identify when changes have been applied to a row or record.
- Use source system transaction files and logs. Be sure that logs are complete, and that transactions that successfully posted can be separated from those that failed.
- Replicate source data changes. Be sure that deletes and transaction backouts are also replicated.
- DBMS logs may be used to identify rows and columns that have changed.
- Compare generations of backup files. This method finds differences between two generations of source system backup files. It offers complete change detection, but acquisition frequency is limited to the frequency at which backups are taken, limiting its utility to high-latency data.
- Use middleware and/or a message broker to push data changes that occur in the source system to the data integration system..

DETECTION BY THE INTEGRATION SYSTEM

Generally applicable only to high-latency warehousing data of relatively small size, it is practical to detect data changes after data has been extracted. Two techniques are common here – comparing generations of extract files, and comparing a previous extract to the current state of the source data. Both methods depend on full data extracts and batch processes to detect data changes.

Data Capture Design and Specification

Messaging



Data Capture Design and Specification

Messaging

MESSAGING DESIGN CONSIDERATIONS

When designing to capture data through messaging consider each of the following:

- Messaging protocol – What standards will you apply to format, structure, and content of messages? XML is widely recognized as a highly flexible and adaptable messaging standard. But with flexibility and adaptability comes the need to establish additional standards. Many XML-based vendor products define standards of messaging protocol. Open standards such as Extensible Messaging and Presence Protocol (XMPP) also exist. XMPP is an open protocol for near real-time messaging.

- Technology– Will you use messaging middleware on an application-by-application basis or implement EAI technology to enable inter-application communications as well as message-based data capture?

- Message queue management – How long will messages be kept in the message queue? What organizations and processes will be responsible to manage the queue? Will removed messages be archived or simply purged? What publish and subscribe capabilities will you have?

- Extract complexity – How difficult will it be to interpret messages and parse out needed data and metadata for integration?

- Application failure and rerun – When a message-publishing application experiences failure and or rerun, how will those events be reflected in the message queue? How will they be interpreted by the processes that receive messages and transform them into integrated data?

- Application queue completeness – When you depend on applications to publish messages, are there any processes (normal or exception) that do not create messages for all data changes?

- Responsibility to receive messages – Are receiving processes (transformation processes in the case of data integration) responsible to extract messages from the queue, or are messages automatically delivered to an always-on agent in the receiving system?

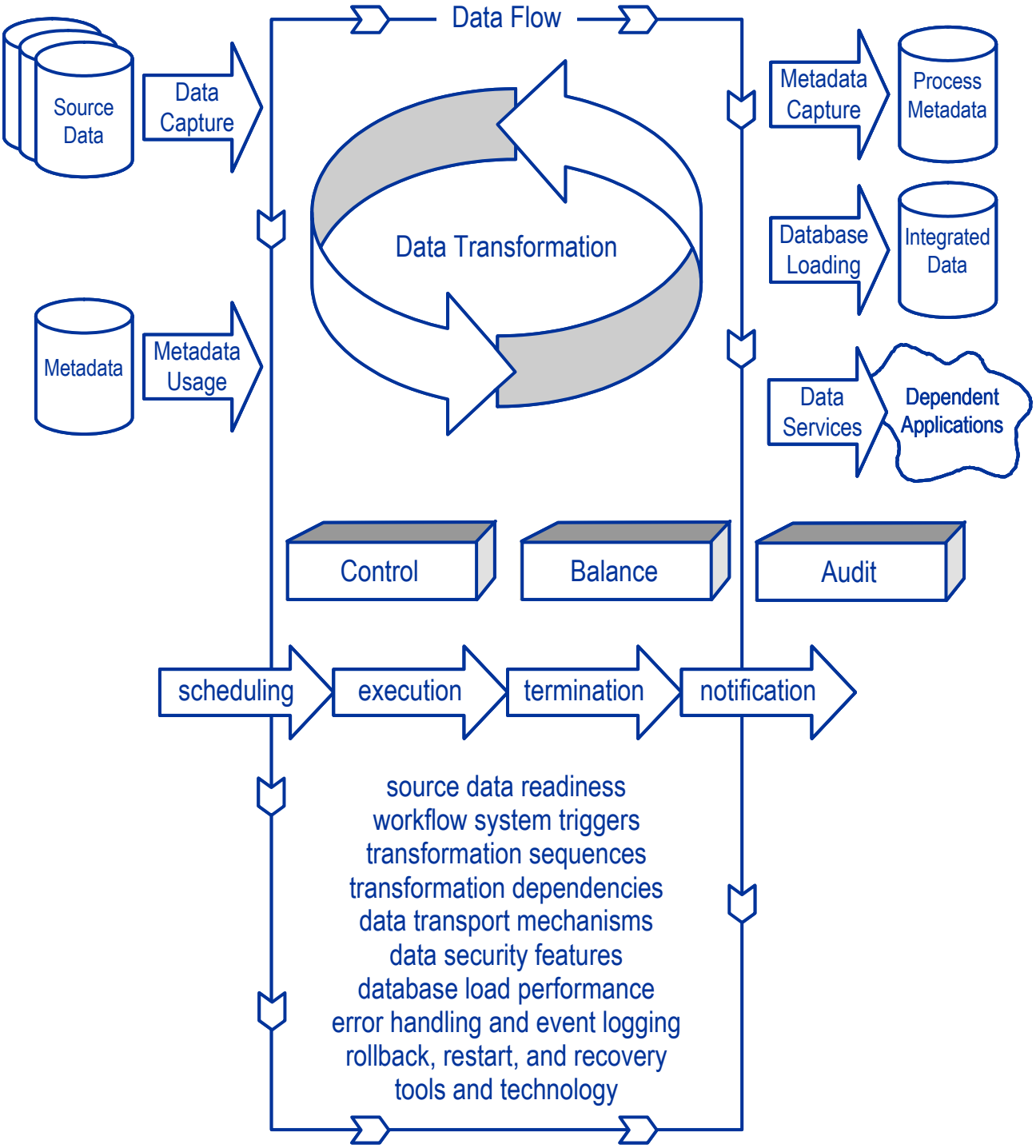
Module 4

Data Integration Technical Design

This page intentionally left blank.

Technical Design Concepts

Comprehensive Processing Design



Technical Design Concepts

Comprehensive Processing Design

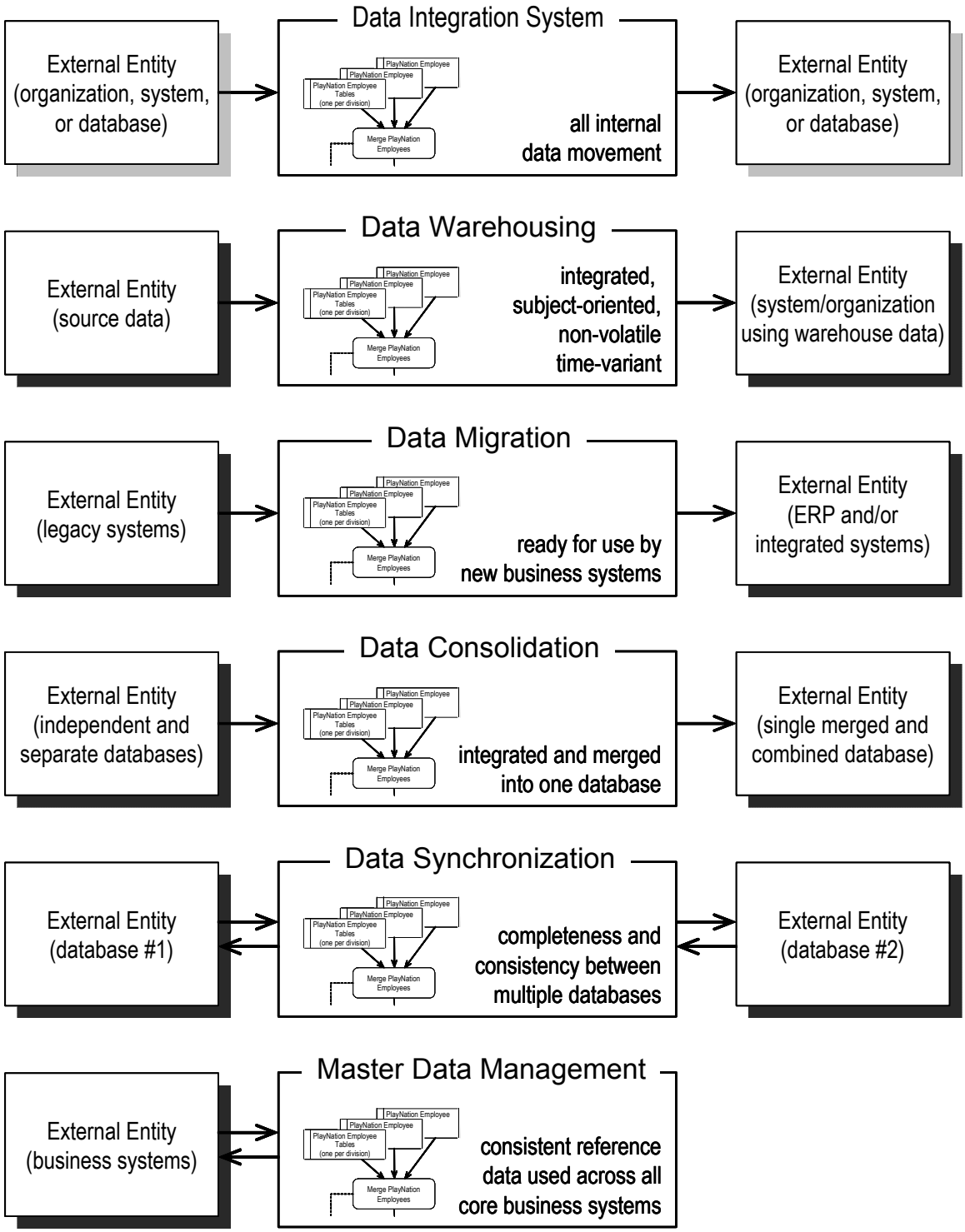
AN OVERVIEW OF DESIGN ISSUES

Technical design extends the functional design to describe how each function is to be implemented and how all of the parts fit together into a cohesive data integration system. Complete design demands attention to all of:

- Source Data Capture – How will data be captured from each and every source? At what frequency? Using what technology? To place the captured data where? And in what format?
- Data Flow – How does data move through the pipeline from disintegrated source data to integrated target data? What processes are involved? What are the inputs and the outputs of each process? Where is data stored? Where are temporary process-to-process files used?
- Work Flow – How is each processing sequence initiated? How are process-to-process dependencies implemented? How is process scheduling implemented? How are dependencies on source system schedules handled?
- Transformation Sequence and Dependencies – How are sequence dependencies among transformation rules implemented? How are multiple transformations packaged as modules and processes?
- Metadata Capture – How will metadata be captured during processing? Where will it be stored? In what forms will it be stored?
- Database Loading – How will database loads be performed? How will referential integrity be assured? How is database indexing managed?
- Process Scheduling – How are processes grouped as scheduled sequences of work? How will process scheduling be implemented?
- Process Execution – How will processing be executed? What steps are needed at the start and end of each process sequence?
- Process Termination – How is normal end-of-processing confirmed and communicated? How is abnormal end-of-processing recognized and communicated?
- Rollback, Restart, and Recovery – How are processing errors and exceptions handled? How will databases be rolled back to a previous state when needed? How will interrupted processing be restarted? How will recovery from processing failure occur?
- Event Logging – How will significant processing events be recorded?
- Error Handling – How will non-fatal errors be reported and repaired?
- Notification and Communication – How will dependent systems and people be informed of the state of processing?
- Audit, Balance, and Control – How will ABC's be implemented?

Data Flow Design

End-to-End Data Flow



Data Flow Design

End-to-End Data Flow

COMPLETE DATA FLOW DESIGN

A complete data flow design for a data integration system always begins with one or more external entities (organizations, systems, or databases that provide inputs to the data integration system) and ends with one or more external entities (organizations, systems, or databases that receive results from the data integration system).

THE BEGINNING

External entities that provide input to data integration systems are typically:

- data sources for data warehousing systems
- legacy systems for data migration systems
- independent systems or databases for data consolidation systems
- independent databases for data synchronization systems (the same databases that receive results from the system)
- business systems for MDM (the same business systems that receive results from the system)

THE MIDDLE

Between the external entities that provide input and those that receive results, there exists a network of all of the processes, data flows, and data stores that are needed to satisfy the integration requirements. It is this network that is decomposed into transformation processes, transformation steps, and transformation rules.

THE END

External entities that receive results from data integration systems are typically:

- systems and organizations that use warehousing data, including business intelligence systems for data warehousing
- ERP or other modernized and integrated systems for data migration
- merged and combined databases for data consolidation
- independent databases for data synchronization systems (the same databases that provide input to the system)
- business systems for MDM (the same business systems that provide input to the system)

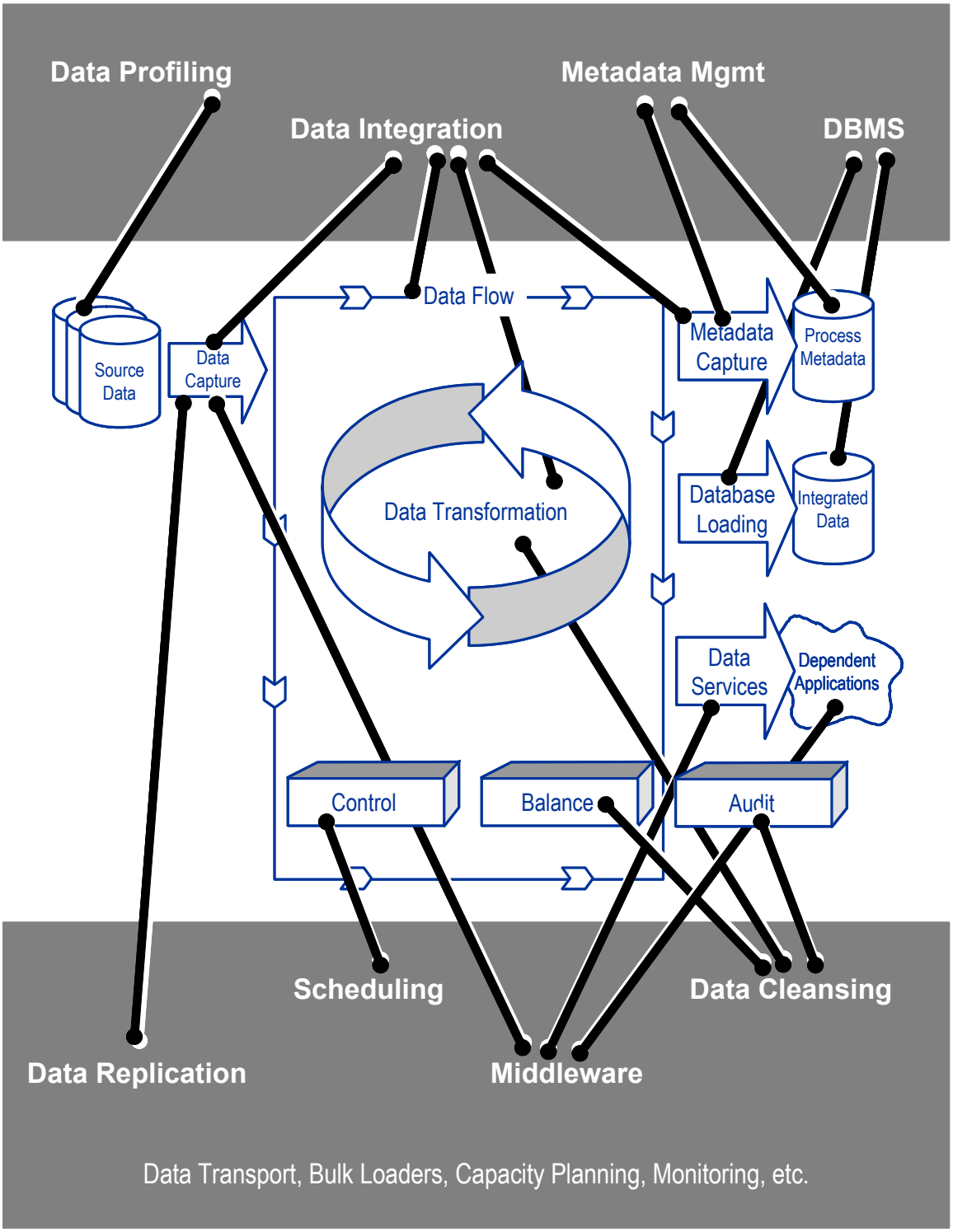
Module 5

Construction, Deployment, and Operation

This page intentionally left blank.

Building Data Integration Systems

Tools and Technology



Building Data Integration Systems

Tools and Technology

TECHNOLOGY DEPENDENT

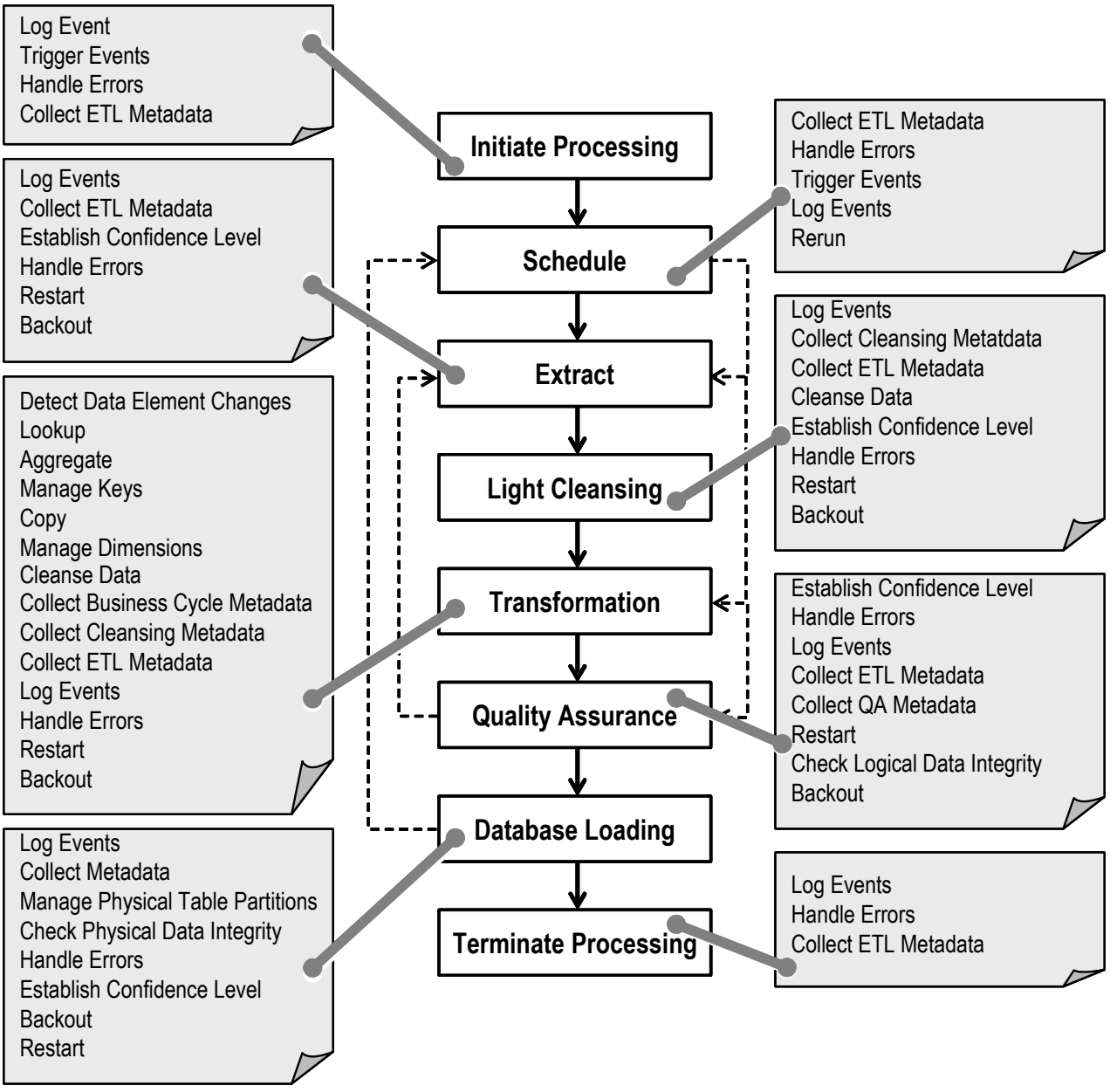
Many different tools and technologies fulfill various roles in data integration systems. Any implementation is certain to depend on multiple technologies to meet its objectives. Some technologies are likely to be pre-determined – the DBMS, for example, in a data migration project. Others may be open to choice. For most types of technology you'll find a broad range of products that vary widely both in capabilities and in cost. Technology changes rapidly. For an up-to-date look at data integration technology visit www.tdwi.org/marketplace.

TECHNOLOGY INDEPENDENT

Although technical design is strongly influenced by technology, and implementation is heavily technology dependent, functional design should remain free of dependency on any particular products or technologies. Especially for ongoing integration systems – data warehousing, data synchronization, and MDM – it is important that the system be able to adapt as technology changes.

Building Data Integration Systems

Standards, Frameworks, Templates, and Reuse



adapted from the ETL Framework and Component Architecture of the University of Washington

Building Data Integration Systems

Standards, Frameworks, Templates, and Reuse

STANDARDS

System development standards, whether for data integration or for other kinds of systems, provide experienced and novice systems analysts, project managers, and computer programmers with guidelines for design and implementation. From lifecycles and methodologies to guidelines for roles and uses of technology, good standards accelerate development and improve the quality of the systems developed.

For data integration projects that are part of a broader program of incremental development – data warehousing and MDM– standards are especially important. Frameworks and templates, in particular, establish integration architecture that is important to incremental development and enable reuse to enhance the speed, quality, and consistency of the systems that are developed.

FRAMEWORKS, TEMPLATES, AND REUSE

A data integration framework simplifies design and development by identifying a standard set of data integration components, describing their roles and interrelationships, and assigning responsibilities to each. When the set of assigned responsibilities encompasses data unification, metadata collection, audit trails and balancing, error and exception handling, and process management it becomes much easier to design and develop a robust and fully-functional data integration system.

Templates take the next step by providing a limited level of reuse. A template is a skeleton set of logic, script, or code that represents the structure and always-present activities of an integration component.

Reuse at the next level develops fully-functional components that are invoked through application program interface (API) or other means. Reusable components range from individual data transformation rules to common functions within data integration sequences, and occasionally to fully-functional transformation steps. Reusable components shift the focus of system construction from programming to assembly

AN EXAMPLE

The diagram on the facing page illustrates a data integration framework and lists many of the reusable components in use at the University of Washington. This framework is applied in a data warehousing program that achieves integration primarily through ETL processing to deliver a hub data warehouse with dependent data marts.

Module 6

Summary and Conclusion

This page intentionally left blank.

Best Practices in Data Integration

Learned through Experience

11. TRANSFORM DATA TO MEET SPECIFIED REQUIREMENTS

SUBJECT ORIENTATION
DATA CONSOLIDATION: ENTITIES, IDENTITY, RELATIONSHIPS, ATTRIBUTES, AND VALUES
LEVEL OF DETAIL
DATA QUALITY

12. CHOOSE KEY MANAGEMENT METHODS CAREFULLY

NATURAL KEYS
SYSTEM KEYS
SURROGATE KEYS

14. DEFINE END-TO-END DATA FLOW

BEGIN WITH AN EXTERNAL ENTITY
IDENTIFY ALL DATA MOVEMENT
IDENTIFY ALL DATA STORES
END WITH AN EXTERNAL ENTITY

14. EXTEND DATA FLOW TO COMPLETE TECHNICAL DESIGN

WORKFLOW
SERVICE LEVELS
PROCESS MANAGEMENT

15. RECOGNIZE THE FULL RANGE OF TECHNOLOGY

DATA PROFILING
DATA INTEGRATION
DATA CLEANSING
METADATA MANAGEMENT
DATABASE MANAGEMENT
DATA REPLICATION
SCHEDULING
MIDDLEWARE
AND MORE ...

Best Practices in Data Integration

Learned through Experience

16. DEFINE DATA INTEGRATION STANDARDS

FRAMEWORKS
TEMPLATES
REUSE

17. INCLUDE SYSTEM MANAGEMENT CAPABILITIES

VERSION CONTROL
RELEASE MANAGEMENT
ERROR TRACKING
PROBLEM RESOLUTION
CONFIGURATION MANAGEMENT
INFRASTRUCTURE MANAGEMENT

18. TEST THE INTEGRATION SYSTEM AT MULTIPLE LEVELS

UNIT TESTING
STREAM TESTING
CYCLE TESTING
ABC;S TESTING

19. IMPLEMENT AS A FULL-STRENGTH PRODUCTION SYSTEM

FORMAL PRODUCTION ENVIRONMENT
SEPARATE DEVELOPMENT AND MAINTENANCE ENVIRONMENTS
DISASTER RECOVERY / BUSINESS RESUMPTION PLANS
ACCEPTANCE TESTING AND END-USER VERIFICATION

20. RECOGNIZE OPERATIONS NEEDS

SYSTEM MONITORING
GROWTH MANAGEMENT AND CAPACITY PLANNING
CHANGE MANAGEMENT
INFRASTRUCTURE SUPPORT
END-USER SUPPORT AND SERVICES

