Ministry of Education

Data Architecture Standards

Version 1.2

May 17, 2016

Information Security Classification : Low

Table of Contents

Introduction **4**

Background **4**

Assumptions **5**

Benefits **6**

Scope **6**

Objectives **7**

Data Architecture Process **8**

Pre-project Phase and project management phase 8

Business Requirements phase 8

Architecture/Design phase 8

Development/Build/coding phase 9

Implementation phase 9

Maintenance phase 9

QCIL Review of Data Architecture work products 9

Data Architecture Standards **10**

Data Model formats 10

Conceptual Data models 10

Logical Data models 11

Physical Data models 11

DDL scripts 11

Modelling Notation 12

Model normalization 12

Model documentation 13

Data Definition reports 13

Logical and Physical Data models synchronization 13

Registries (MASTER DATA repository) 14

Metadata Standards 14

Other Standards 14

Industry best practices for modelling 15

Data Architecture Controls **15**

Configuration Management 15

Release Management 15
Database Backup and Recovery 16
Data Retention and purging 16
Data Architecture Standards for COTS, SaaS and OpenSource 16
Model naming, Model labeling and Model generation standards 17
Model File Name 17
Model Labeling 17
Revision Log 18

Introduction

The purpose of this document is to provide consolidated Data Architecture standards and guidelines for the Ministry applications during application development, implementation and maintenance phases. The standards and guidelines described in this document are encompassing and supplementing the already existing data modelling standards (such as "*Requirements Modeling and Specification Guidelines and Standards*" and "*Domain Class Diagram Modeling Standards & Guidelines*" etc on the BPP site).

The standards and guidelines described in this document are generic enough to be applicable to various types of application development and maintenance projects in the Ministry as well as non-application projects/initiatives that may have data modelling tasks.

Background

In the past, *Oracle Designer* used to be the Ministry standard tool for all Data Architecture work (modeling of logical and physical entities, model validations, model reports generation etc). Some projects have also used the *Oracle Headstart* tool for model validations. All the Data Architecture standards were defined in and referenced from the *Oracle Designer* directly. In addition, the "*Requirements Modeling and Specification Guidelines and Standards*" document (available on BPP site) provides the standards for the data modeling during the Business requirements phase.

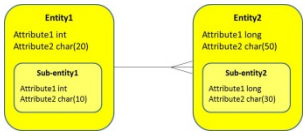
- The Oracle Designer and the Oracle HeadStart tools have been **deprecated** by the Ministry few years ago due to the de-supporting of the tools by the Oracle Corporation. Some ministries are using *Sparx Enterprise Architect* as the standard tool for data modelling and repository purposes.

- Ministry of Education has two desktop licenses of *Sparx Enterprise Architect 10*.

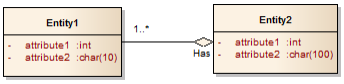
- Standards and guidelines specific to "Data modelling" ("data modelling" is a subset of overall Data Architecture process) are already existing and available on the BPP sharepoint site, as described below.

- The "*Requirements Modeling and Specification Guidelines and Standards*" document describes the standards for Entity Relationship Diagram (ERD) modelling using the E-R notation (crow-foot notation).

Example :



– The "*Domain Class Diagram Modeling Standards & Guidelines*" document describes the standards for UML class diagram notation for drawing the Domain class diagram etc.
Example :



See later sections of this document for more information on the standards for modeling notation. [New development projects need to use E-R notation (crow-foot notation) for data modelling. New UML-centric projects may use UML class diagram notation for data modelling instead of E-R crowfoot notation, subject to prior approval by Ministry Architecture Committee (MAC). Existing applications may continue with their current notation (E-R notation, UML notation (if approved by MAC), whichever is the case).]

Assumptions

The following are some high-level assumptions with regard to the Data Architecture standards of the Ministry :

- Data Architecture standards (defined in this document and elsewhere on BPP site) are part of the overall Business Program Planning (BPP) standards of the Ministry.

- All BPP standards (and the Data Architecture standards thereof) are owned by the Ministry Architecture Committee (MAC).

- Any changes to the Data Architecture standards are to be reviewed and approved by MAC prior to the publication and usage of the revised standards.

- All changes to the Data Architecture standards (and other standards on the BPP site) are implemented through the BPP change management process (which is different from the Application change management process known as "CR" process owned by the Ministry's Change Advisory Board (CAB)) . All "requests for BPP standards change" are to be forwarded to the Application Architect to facilitate the analysis, MAC review and implementation of the requested change.

- Following are the options for the *data modeling tools* and storage of models in the tool repository :

  - *Oracle Designer 10g (**deprecated**)*. The tool is configured with Oracle database as default. NOTE : *Designer must not be used for new development projects*).

  - *Sparx Enterprise Architect*. Oracle needs to be configured explicitly as a database for the tool. By default, the data models are stored locally as ".eap" files in metadata (xml) format, unless a backend database like Oracle is configured with the tool. NOTE : *UML notation for data modeling is to be used only for UML-centric projects, subject obtaining prior approval from Ministry Architecture Committee (MAC).*

  The above tools also support the generation of graphical/image copies of data models (".jpg" etc).

See later sections of this document for standards on data modeling tools, storage of models etc.

Benefits

The following are some high-level benefits with regard to the Data Architecture standards of the Ministry :

- Consistency across Ministry for all applications/initiatives that have Data Architecture tasks and deliverables
- Better traceability of Data Architecture deliverables and artefacts
- Provides better clarity to the architects and developers in solution building
- Provides better abstractions (perspectives from generic to abstract to specific) on data architecture models

Scope

In the context of Data Architecture standards and the overall BPP standards, any business software having the following characteristics is called an "*Application*". All those business

software (referred as “applications” hereinafter) need to follow the Data Architecture standards as well as other BPP standards , for new development as well as for application changes implementation projects :

- application that is custom-built (in-house built or built by external contractor) using current technologies or legacy technologies (or the combination) and
- hosted either on the BC Government network (SSBC) or on third party sites outside the BC Government network and
- used by Ministry of Education staff and/or
- use by external clients such as School Districts, Schools and/or
- used by public (such as students, parents, any other public etc) and/or
- used by external business partners/contractors (such as AMS service providers etc)

Data Architecture standards are also applicable for the COTS products/solutions, Software as a service (SaaS) and Open Source solutions. These are described in separate section in this document.

Data Architecture standards are also applicable for various other types of applications such as the dynamic web sites, eServices applications, desktop applications (e.g. Excel workbooks, MS Access databases) etc. These are described in separate section in this document.

Objectives

The objectives of the Data Architecture standards are :

- To enable consistency in the data models and data architecture practices across the various application development and maintenance projects/initiatives in the Ministry.

- To enable traceability of the data models, artefacts and data definitions.

- To enable cross-references between the data models, artefacts and data definitions for any given application and the dependent applications/interfaces.

Data Architecture Process

The process of Data Architecture for an application development project or application change project should start at early stages in the project, typically starting at the Business Requirements phase.

Pre-project Phase and project management phase

It is unlikely that any significant Data Architecture work is involved during the pre-project phase (during *business case*, *options analysis decision note* etc) or the project management phase (during *Project charter*, *Master Project Plan* etc). In case it does, it is recommended to follow the Data Architecture standards. It is also recommended to preserve all Data Architecture work products (deliverables and artefacts etc) for future reference and reuse purposes during the later phases of the project.

Business Requirements phase

The Data Architecture work during Business requirements phase is usually done at a logical level (conceptual level) to ensure the Business Requirements are adequately describing the various "data" requirements of the application. The "*Requirements Modeling & Specification Guidelines & Standards*" document available on the BPP site needs to be referenced for Business Requirements modeling (E-R notation). In addition, the "*Domain Class Diagram Modeling Standards & Guidelines*" document available on the BPP site needs to be referenced for UML class diagram notation. NOTE : The E-R notation (crowfoot notation) needs to be used for data modeling. UML-centric projects may use UML class diagram notation (instead of E-R (crowfoot) notation), subject to prior approval by Ministry Architecture Committee (MAC). Existing applications may continue with their current notation (E-R notation or UML notation (if approved by MAC), whichever is the case).

It is highly recommended to preserve all Data Architecture deliverables and artefacts from "Business Requirements phase" for future reference and reuse purposes during the later phases of the project (architecture/design phase, coding phase etc).

Architecture/Design phase

Bulk of the Data Architecture work is done during the Architecture/Design phases (particularly during Data Architecture/Design phase and also to some extent during Application Architecture

Phase). The Data Architecture standards (defined in this document and elsewhere on BPP sharepoint site) must be followed during data architecture/data modelling phases. All Data Architecture work products (such as deliverables, models, diagrams, artefacts, data definition reports etc) are subject to Ministry QCIL review process. All Data Architecture work products must be preserved in the "updateable" format and stored in the Ministry BPP designated storage locations/repositories (currently on PDO Documentation Library on Sharepoint, and also in Oracle Designer repository for many existing applications).

Development/Build/coding phase

Data models may need to be modified during the development/build/coding phase of the application. All modifications to data models must conform to Ministry Data architecture standards. During the development/build/coding phase, any changes to be done to the already approved/signed-off data architectures/data models are subject to QCIL review process again.

Implementation phase

It is unlikely that any Data Architecture work is involved during the Implementation phase. However, if there is any data architecture finding during the implementation phase that needs to be addressed prior to production rollout, such finding are subject to QCIL review process.

Maintenance phase

All data architecture work (logical and physical data model changes etc) during the application maintenance (change implementation) needs follow the Ministry's Data Architecture standards and processes described in this document. All data architecture changes during the application maintenance (change implementation) are subject to QCIL review process.

QCIL Review of Data Architecture work products

All data architecture work products (such deliverables, models, diagrams, artefacts data definition reports etc) during any phase of the application life cycle (requirements, architecture/design, build/code, testing, implementation, maintenance) are subject to the Ministry QCIL review process.

Data Architecture Standards

Data Model formats

The Data Architecture models and diagrams also need to be in metadata format, not just in the graphical (image) format. If only the graphical format is feasible for any justified business/technical reasons, prior MAC approval is to be obtained before commencing the data architecture work in graphical format for the project.

The E-R notation (crowfoot notation) needs to be used for data modeling consistently for all projects. The data models in E-R notation (crowfoot notation) needs to be created using the [1]*Sparx Enterprise Architect* tool and submitted to Ministry in the ".eap" format (metadata format), or as an XMI export file. In addition, the graphical (image) formats of the data models (such as ".jpg" files etc) are also to be to be submitted.

UML-centric projects may submit data models in UML class diagram format (instead of ERD (crowfoot) format), subject to prior approval of UML notation by Ministry Architecture Committee (MAC) for each UML-centric project on case basis. In addition, the [2]graphical (image) formats of the data models (such as ".jpg" files etc) are also to be to be submitted.

Existing applications may continue to maintain their data models in their current tool and follow the tool's standards and formats. In addition, the [1]graphical (image) formats of the data models (such as ".jpg" files etc) are also to be to be submitted.

Regardless of the modeling tool and modeling notation used , all data models in metadata formats (".eap"/."XMI") and in graphical (image) formats (".jpg" etc) as well the associated DDL scripts need to be packaged along with application source objects and uploaded to Ministry Subversion tool. See the section "Configuration Management" later in this document for more information.

Conceptual Data models

Application projects will normally begin their data architecture work in the Business Requirements phase. As part of the Business Requirements Document, a Conceptual Data Model is used to represent the Data Architecture, in the form of Entity Relationship Diagram (ERD). The Conceptual Data Model represents in business terminology the major data entities and relationships which represent the data scope of the project. As this is an early version of the ERD, the initial business entities and data relationships

are depicted to support the business requirements represented in the BRD. At this stage, details for all data attributes are not likely known and 'many to many' relationships will not be fully resolved.

Logical Data models

All Applications regardless of their technologies (current or legacy technologies) must have a Logical data model. The Logical data model (also represented as an ERD), is produced in the Design Phase of the project, as part of the Application Architecture deliverable. In this stage, each required data entity must be included to represent the scope of the project, so that a physical data model can be generated. The Logical data model must be in metadata format in addition to diagram (graphical) format. Logical data model should not be a reverse-engineered version of the physical model or the physical database schema, since such reverse-engineered logical models will not be of any practical value. NOTE : For *COTS, SaaS* and *Open Source* solution implementations, please refer the last section in this document (section "*Data Architecture Standards for COTS, SaaS and OpenSource*").

Physical Data models

All Applications regardless of their technologies (current or legacy technologies) must have a Physical data model. By the end of the Design phase, a Physical data model is created as part of the Application Architecture deliverable. The Physical data model represents the physical table structures in scope of the project. Relationships to existing physical tables should be shown. Color coding can be used to distinguish these. and it must be in metadata format as well as diagram (graphical) format, not just in DDL format. Also, the physical data model should be preferably generated as much as feasible from the corresponding logical data model, not manually created entirely. NOTE : For *COTS, SaaS* and *Open Source* solution implementations, please refer the last section in this document (section "*Data Architecture Standards for COTS, SaaS and OpenSource*").

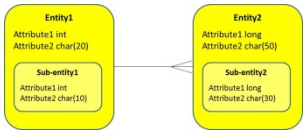
DDL scripts

All Data Definition Language (DDL) scripts (such CREATE TABLE/VIEW statements) should be preferably generated as much as feasible from the corresponding physical data model, not manually created entirely. All DDL scripts need to be packaged along with application source objects and uploaded to Ministry Subversion tool. This is to tie and sync the DDL scripts with the application release and its data models.

Modelling Notation

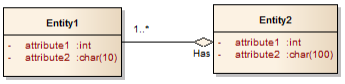
As per the Data Architecture standards, the Ministry accepts data models in two most commonly used industry standard notations :

- The *E-R notation* (crowfoot notation), is the Ministry standard, and is applicable for most Ministry projects. Example :



- For E-R notation (crow-foot notation), the modelling/notation standards defined in the "*Requirements Modeling and Specification Guidelines and Standards*" document on the BPP site are to be followed.

- The *UML class diagram notation*, which is applicable for UML-centric projects (subject to prior MAC approval). Example :



- If UML class diagram notation is approved by MAC for the project, the standards defined in the "*Domain Class Diagram Modeling Standards & Guidelines*" document on the BPP site are to be followed.

- NOTE : Use only one of the notations (E-R or UML, not both) for data modeling purposes for any given application/project. If both notations are to be simultaneously used for data modeling of any project for justified reasons, prior MAC approval needs to be obtained.

Model normalization

For operational OLTP systems, the models/entities/tables must be normalized to at least up to the third normal form ("3NF").

For OLAP systems (Data Warehouse and Business Intelligence), the normalization levels (if any) are determined by the Warehouse/BI specialists based on the nature of the specific query/report.

Model documentation

All Data Architecture work products (logical models, physical models, model elements such as entities/attributes, DDL scripts, database schema objects such as tables/views, columns etc) must be adequately labeled and documented. The Sparx Enterprise Architect tool (and laso most other data modelling tools in the industry) provide the facility to add “comments” for the model elements (entity, attribute etc) and schema elements (table, view, column etc).

Data Definition reports

All types of Applications (custom-built/COTS/Software as a Service (SaaS) applications etc) regardless of their technologies (current or legacy technologies) must have the Data definition reports for logical model and also for physical model. The data definition reports are typically in text format with some embedded diagrams/illustrations for the sake of better clarity.

Most data modelling tools (such as Sparx Enterprise Architect, etc.) provide the facility of generating and printing/saving the data definition reports from the logical models and physical models.

Logical and Physical Data models synchronization

It is most ideal and highly recommended that the physical data models are always entirely generated from the logical models and the DDLs are always entirely generated from the physical models. However, this is not often feasible in reality due to tool limitations and other technical reasons. Hence the logical data model needs to be periodically reviewed and compared with the corresponding physical data model and DDL scripts and all three must be updated and synchronized from time to time.

A disciplined approach to data architecture/data modelling will enable in keeping the model synchronization efforts to minimal levels :

- If the business requirements have changed, a change to the Conceptual Data Model may be required, as part of the BRD changes.

- For design level, changes, always start with logical data model while doing any changes to the data architecture (models, DDL scripts, database objects etc)

- After incorporating the required changes in the logical model, incorporate the same changes in the physical model (or re-generate the affected components of the physical model directly from the logical model)

- After incorporating the required changes in the physical model, incorporate the same changes in the DDL scripts (or re-generate the affected parts of the DDL script from the physical model)

- After applying (running) the DDL scripts in the database schema to create/modify/delete database objects, review all the layers (*logical model*, *physical model*, *DDL script*, and *database objects*) to ensure the required changes have been implemented consistently in all the layers.

Registries (MASTER DATA repository)

Multiple applications may need the same common master data (registry data) , such as list of COUNTRIES, list of PROVINCES etc. It is highly recommended to share and reuse such master data/registry data (and their metadata) and not to re-build such entities/tables in applications.

Metadata Standards

All metadata objects (logical and physical meta-objects and the associated DDL scripts for the meta-objects) must be adequately documented to indicate that these are metadata objects. Governance for the metadata objects and the data therein must be clearly defined and documented during Data Architecture phase of the application.

Other Standards

The Ministry supports having database referential integrity 'turned on' wherever possible, to help ensure data quality and data standardization. Exceptions may be made where appropriate.

Through QCIL reviews, the Ministry architects will ensure that practical designs are proposed and implemented. This may include considerations for performance, data quality, and simplicity. Please discuss data design options with Ministry Architects well in advance, e.g. generalization of data entities.

Industry best practices for modelling

All Data Architecture work products (such as Conceptual, Logical and Physical data models, DDL scripts, database schema objects, etc.) also need to follow industry best practices of data modeling as well, in addition to the Ministry Data Architecture standards.

Data Architecture Controls

Configuration Management

Regardless of the modeling tool and modeling notation used , all data models in metadata formats (".eap" "XMI", etc) and in graphical (image) formats (".jpg" etc) as well the associated DDL scripts and other data modeling artefacts need to be packaged along with the application source objects and uploaded to Ministry Subversion tool. This is to ensure tie and sync the data models with the application release.

Existing applications may continue using their current configuration management methods (such as the Oracle Designer's configuration management system (version control, checkin/checkout etc)) for their data architecture work products (data models, DDL scripts, any other data modeling artefacts, etc.)

Release Management

The implementation (running) and testing of DDL scripts in the DEV and TEST environment is to be done by the developers (typically the service provider staff such as AMS (CGI) staff).

The implementation (running) of DDL scripts in the UAT, PROD and EFIX environment is done by the Ministry DBAs. Only DBAs have access to these environments.

Database Backup and Recovery

As per BC Government standards.

Data Retention and purging

As per BC Government standards. Individual applications may have specific data retention and data purging requirements and frequencies and these needs to be documented during Data Architecture phase (as well as in the Business Requirements phase).

Data Architecture Standards for COTS, SaaS and OpenSource

The following are the Ministry data architecture standards for Commercial Of the Shelf (*COTS*), Software as a Service (*SaaS*) and *OpenSource* implementations (regardless of hosted on third-party sites or on Ministry infrastructure) :

1. Logical data models (whole model and part models) need to be available on demand.
2. Logical Data Model descriptions (entity/attribute descriptions) needs to be available on demand.
3. If the application/solution is hosted on the Ministry servers, then the Ministry technical architecture standards needs to be followed.
4. If the logical models are to be uploaded to Ministry standard tools/repositories for any reason, then the standards and processes of that tool/repository are to be followed.
5. If the COTS/SaaS/OpenSource implementation offers additional adhoc query/reporting facility, such facility is to be used in accordance with Ministry BPP and Technical architecture standards.
6. If the COTS/SaaS/OpenSource implementation needs system integration with other existing application(s), such system integration facility is to be used in accordance with Ministry BPP and Technical architecture standards.

Model naming, Model labeling and Model generation standards

Model File Name

<<*application-acronym*>>_<<*model-type* >>Model**.**<<*file-extension*>> , where :

application-acronym = Application Acronym such as SDE, TRAX etc.
model-type = Conceptual, Logical, Physical, Reversed etc.
file-extension = .eap, .jpg etc (depending on the format of the model file)

Examples

SDE_ConceptualModel.eap , SDE_ConceptualModel.jpg
SDE_LogicalModel.eap , SDE_LogicalModel.jpg
SDE_PhysicalModel.eap , SDE_PhysicalModel.jpg
TRAX_ReversedModel.eap , TRAX_ReversedModel.jpg

It is recommended to use the same name for the generated model file and the name of the model inside the tool. This will help in consistency and traceability.

Model Labeling

All models (generated files as well as inside the tool) need to have the following labels :

- Model version (##.##) (Follow same version number as the application in Subversion)
- date last updated (Provide a brief summary of updates/changes done)
- date created
- legends (if any) (example : entities with yellow background are 'super-types')

Revision Log

Date	*Version*	*Change Reference*	*Reviewed by*
June 9, 2015	1.1	Added "SQL Developer" tool as the Ministry standard tool for data modeling.	Ministry Architecture Committee (MAC)
June 9, 2015	1.1	Made relevant changes to various sections in view of the above.	Ministry Architecture Committee (MAC)
June 9, 2015	1.1	Added a new section for "Model naming and Model labeling standards" in view of the above.	Ministry Architecture Committee (MAC)
May 17, 2016	1.2	Removed references to SQL Developer tool due to Sparx EA tool becoming Ministry standard.	Ministry Architecture Committee (MAC)
May 17, 2016	1.2	Added additional sections : • Conceptual data Model	Ministry Architecture Committee (MAC)