Quality management reference models for Business Intelligence - class systems

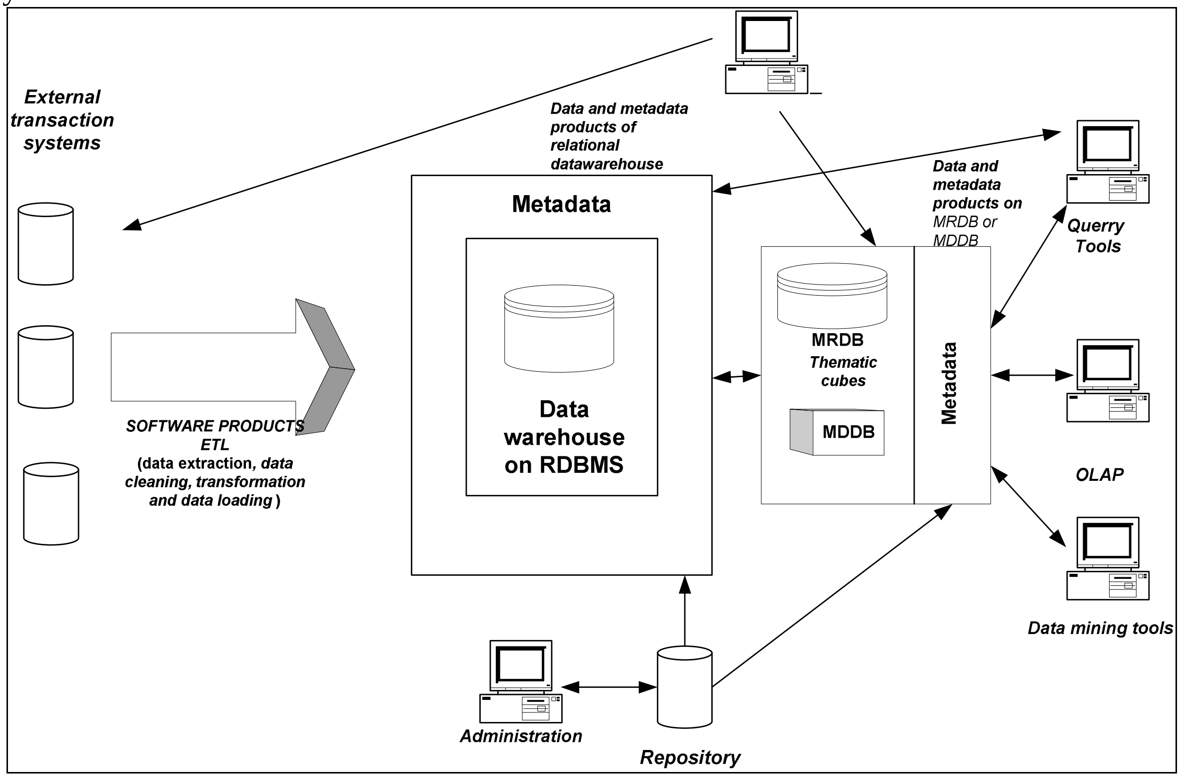
Ph.D. Jerzy ROSZKOWSKI[1]
Prof. Ph.D. Andrzej KOBYLIŃSKI [2]

Introduction

The issue of software quality in Business Intelligence (BI) systems presents challenges of far greater complexity than in the case of typical software applications the code of which software developers are most commonly tasked with writing. In the latter case, quality evaluation can be based on the quality of the resulting software product itself. Products designed for Business Intelligence class systems, however, are much more complex in character as they include not only software products, but also final products (analytical reports), which are created in codeless mode. In both instances, the quality of final products is also influenced by the quality of intermediate products (analysis and project documentation), the creation of which is usually aided through the use of CASE tools. The purpose of the present article is to demonstrate the results of research projects which endeavoured to evaluate the quality of BI products by constructing reference models. The endeavours referred to above were made in cooperation with the architects of BI-class solutions.

1. The architecture of Business Intelligence-class systems

The products earmarked for this class of systems flows directly from the architecture of the given system. Based on the four-layer architecture presented below, the following groups of products for BI systems can be identified.



---

1 Management Systems Consulting, 93-134 Łódź, Poznańska 28/1, jerzy.roszkowski@neostrada.pl
2 Department of Business Informatics, Warsaw School of Economics (SGH), andrzej.kobylinski@sgh.waw.pl

Figure 1. Overview of BI-class system architecture. Source: author's study

- **ETL software products[3] (extraction, cleansing, transformation and loading).**

A important stage of data warehouse (DW) implementation is the extraction of data from transaction processing systems and the conversion thereof into formats supported by the given DW in its analytical capacity. The extraction of data from transaction processing databases for the purposes of transferring such data into a data warehouse requires such data to be cleansed and transformed. This in turn requires either the use of specialist tools or the development of dedicated written software to manage metadata and transform separate data into information which may later be used by decision support tools. The tools in question need to have the following functionality:

- removal of unwanted data from transaction processing databases,
- conversion of data and definitions into common designations,
- calculation of summary data and derived data from source data (functional transformation),
- determination of estimated values for missing data,
- adaptation of source data for redefinition .

The operation of tools designed for data retrieval, cleansing, extraction, transformation and migration may be impeded by external circumstances such as:

- *database heterogeniety*- individual database management systems (DMBS) are based upon different data models (hierarchical model, network model, relational model), different languages used for database access, data navigation and data manipulation, different integrity conditions or database recovery conditions,
- *data heterogeniety*- homonymity and synonymity of data designations, different scope of attributes applied to identical objects in different transaction processing systems, different modelling methods for identical facts.

- **Metadata-related products** (metadata for relational, RDBMS-based data warehouse and metadata for MDDB multidimensional database or MRDB multirelational database).

Metadata contains descriptions for data contained in data warehouses and are used for data warehouse creation, maintenance and management. The repository contains:

- source data description,
- data transformation description, i.e. source data mapping from transaction management systems to the data warehouse,
- data conversion algorithms,
- data warehouse object description and data structure description,
- history log for warehouse data usage and extraction.

Metadata facilitates interactive access into stored data so that the contents thereof may be comprehended or so that the desired fragments of data can be identified. Metadata management is conducted by means of a metadata repository. The software which manages the repository is used to map source data into the initial database, to generate the code necessary for data transformation and to oversee the transfer of data into data warehouses. The software referred to above typically runs on workstations, which allows the user to determine how data is to be transformed, mapped, converted and summed. Metadata management constitutes a critical aspect of data warehouse technology. For some data warehouse architecture solutions, the metadata repository is referred to

3 ETL (Extract, Transform and Load)- a set of tools facilitating the process wherein data is obtained for data warehouses

as the “*data dictionary*” or the “*encyclopedia*”.

- **Data-related products** (data warehouse databases)

In most cases, a data warehouse is based upon a relational database management system (RDBMS). Traditional RDBMS technology has its limits, however, due to the fact that these databases are optimized for transaction processing. Some data warehouse attributes, such as:

- substantial size of the database,
- processing of complex queries,
- need for flexibility in relation to varied user requirements (data mining, multiple table merging, aggregate data viewing modes),

require a different approach to data warehouse databases. This approach includes:

- parallel query processing,
- using indexing structures to increase processing speed,
- multidimensional databases (MDDB) combined with online analytical processing (OLAP), or relational databases designed to support multidimensional features (so-called multirelational databases- MRDB).

- **Reporting product generation tools, query generation tools and data mining tools.**

The most basic purpose of data warehouses is to provide business users with information in order to facilitate their decision-making process. Users take advantage of the data warehouse by means of the so-called front-end tools. These tools may be divided into following categories:

- *Reporting and query tools*

These tools utilize SQL for query management aided by a graphic interface. The most common response to user queries takes the form of pre-formatted reports. The more complex the queries become, the less useful these tools are.

- *Analytical processing tools (OLAP)*

This type of tools is based on the multidimensional database concept, using complex, multidimensional “views” for its analytical purposes. These tools are designed for the most advanced of users. The assumption is that databases for these tools are organized into multidimensional models, which is supported either by a special multidimensional database (MDDB) or by a relational database designed to support multidimensional features (so-called multirelational databases- MRDB).

- *Data mining tools*

These tools are used to identify previously unknown facts (based on inferential rules) or interconnections between various data.

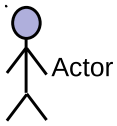
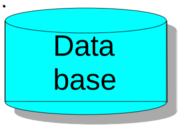
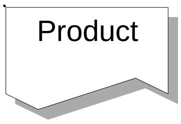
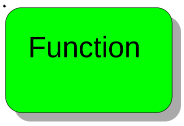
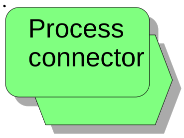
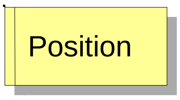
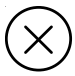
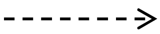
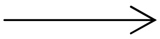
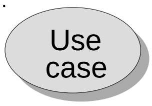
2. Referential modelling definitions

The following definitions of graphic symbols used in referential models are derived from ARIS[4] and UML[5] methodology.

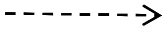
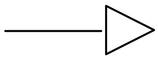
4 ARIS (Architecture of Integrated Information Systems)- methodology and tools developed by IDS Scheer company.

5 UML (Unified Modelling Language)- a formal language used to describe the world of objects for the purposes of

Table 1. Definitions of graphical symbols used in referential models

Concept	Graphical symbol	Semantic meaning
Actor		An object being a person, an organizational unit or another system which interacts with the system in question by means of a Use Case
Database		A collection of data managed by a Database Management System
Product		The end result of a software creative process (most commonly a document)
Business Function		A sequence of activities performed in order to produce an end result (a product or a service)
Interface (connector from/to process)		A connector leading to another model representing a complex process
Organization unit/position		The symbol of a unit in an organization which performs a specific role within the system
Logical operator (AND)		Logical operator “AND” connecting two logical objects
Logical operator “XOR”		Logical operator “OR” connecting two logical objects
Control transmission		A graphical symbol connecting business functions which indicates the order of performance thereof
Data flow		An arrow indicating the flow of data/documents between business functions
Use Case (business)		An elementary business function, by means of which an Actor interacts with the system

object analysis and object programming, officially defined by the Object Management Group (OMG).

Relation “Extend”		A relation connecting two Use Cases, one of which forms an extension of the other
Relation “Generalisation”		A relation connecting two Use Cases, one of which (indicated by the arrow head) is a generalisation of the other
Event		Anything that changes the condition of the system. Business functions must always be divided by events.

3. Development process models

It is recommended that the iterative character of the data warehouse development process model, as well as the development process based thereupon, should be close to the recommended RUP[6], i.e. :

- The procedure is repeatable for each packet of reports representing a specific business area;
- The development cycle procedure may interact with the change management procedure (entry into development cycle procedure) and the testing procedure (exit from development cycle procedure).

Figure no. 2 below demonstrates a model of such a development process, while table no. 2 contains the description of business processes for a single iteration which corresponds to the considered model.

6 RUP (Rational Unified Process)- a process for iterative software development created by Rational Software Corporation.

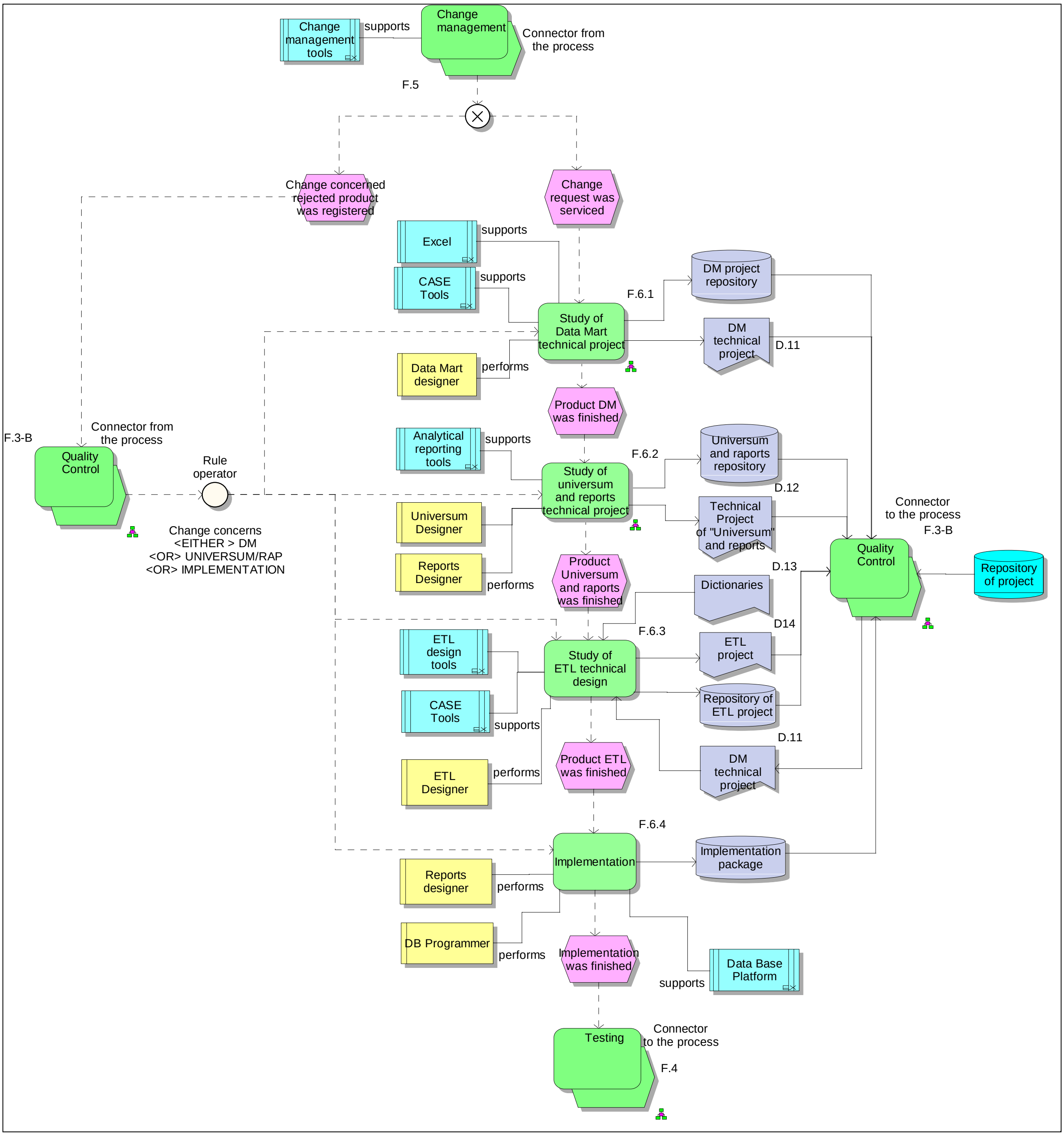


Figure no. 2. General referential model of a development cycle for a BI-class system (eEpc diagram executed in accordance with ARIS standards). Please note that acronyms used in the diagram are explained in the text. Source: author's research.

➢ Business processes within the development process

The description of the business processes for the model shown on Figure no. 2 are contained within table no. 2.

Table no. 2. Main business processes and development cycle processes for a BI-class system.

No.	Identifier in Figure no. 2	Name of business function/description	Frequency
Main processes			
1.	F.3	**Quality Management**. External in relation to process F6. The person responsible for the process is the Quality Manager. Reversion to the construction process of the same product shall take place if the Quality Manager rejects the process. Process F.3 consists from and F.3-A (**Quality requirement definition processes)** and F.3-B (**Quality Control**)	Initiated every time a product is created.
2.	F.4	**Testing**. External in relation to process F6: development cycle ("Development"). When process F4 is initiated, termination of process F.6 takes place.	Initiated each time when a product or a new version thereof is created.
3.	F.5	**Change management**. External in relation to process F6: development cycle ("Development"). The registration of a change in process F.5 constitutes a triggering event which initiates process F.6.	Iterative- modified after change is introduced.
4.	F.6	**Construction**. Main development process.	
Construction (Development cycle processes)- (Figure no. 2)			
5	F.6.1	**"Data Mart" technical design development**. The first of F.6 sub-processes. Should be supported by means of CASE[7] tools (e.g. Enterprise Architect). The person responsible for this process is the Data Mart Designer. The result of this process is the "Data Mart Technical Design" product.	Once, optional for each pack or modification after change is introduced.
6.	F.6.2	**"Universum" and reports technical design development**. This process follows directly after F.6.1 and should be supported by means of analytical reporting tools (e.g. Business Objects). The person responsible for this process is the "Universum" and Report Designer. The result of this process is the "Objects World and Reports Technical Design" product.	Once for each pack, each time after change is introduced
7.	F.6.3	**ETL technical design development**. This process follows directly after F.6.2 and is supported by means of a data mapping tool (e.g. Excel) and a CASE tool (e.g. Enterprise Architect).The person responsible for this process is the ETL Designer. The result of this process is the "ETL Technical Design" product.	Once after its creation or the creation of e new version thereof.
8.	F.6.4	**Implementation**. This process follows directly after F.6.3 and is supported by means of a database implementation platform. Through this process an implementation pack is created, consisting of both code and separate procedures governing the order of its execution in a software tool environment.	Each time after a new version of a given pack is completed.

Source: author's research.

The products of the development process are:

- The Data Mart Technical Design,
- The Object World and Reports Technical Design,
- The ETL Technical Design,
- The Implementation Pack,

[7] CASE -Computer-Aided Software Engineering

The following participants of the development process (known as actors under UML terminology) take part in the creation of the above products:

- The Data Mart Designer,
- The “Universum” and Reports Designer,
- The ETL Designer,
- The DB Programmer.

Table no. 3 demonstrates the general specification of the functions of each of the participants of the development cycle.

Table no. 3. Activities/tasks of the Construction (development cycle) participants; Activities/tasks for the BI-class system quality control process.

No.	Process/ activity number	Activities/tasks	Who performs the activity?	Notes
Activities and tasks for the BI development cycle – process F.6 (figure no. 2)				
1.	F.6.1	Data Mart Technical Design development.	Data Mart Designer	
2.	F.6.2	“Universum” (Object World) and Reports Technical Design development.	“Universum” and Reports Designer	
3.	F.6.3	ETL Technical Design development.	ETL Designer	
4.	F.6.4	Implementation.	DB Programmer	
Activities and for the Quality Control process F.3-A(figure no. 4)				
5.	M.01	Submission of a complete product (following its registration in the products database) for quality control.	Project Phase Leader	E.g. e-mail
6.	F.3.5	Quality Control.	Quality Manager	
7.	F.3.5.1	Product Status Change.	Quality Manager	Only where quality requirements are met
8.	F.3.6	Preparation of a list of remarks and/or quality control protocol.	Quality Manager	
9.	M.02	Notification of the Project Leader concerning the completion of quality control process and deadlines for product improvements.	Quality Manager	E.g. E-mail; Only in the case of failure to meet quality requirements

Source: author's research.

4. BI-class system development process quality models (based on product quality)

As indicated before, final software products do not form the entirety of BI-class system development output. Such products for the process shown in figure no. 2 include the following intermediate products:

- Data Mart Technical Design,
- “Universum” (Object World) Technical Design, reports,
- ETL Technical Design.

The only software product of the process referred to above is the implementation pack. The quality of this product, however, is dependent upon the quality of intermediate products referred to above since those products come into existence prior to the completion of the implementation pack. In accordance with the majority of project management process methodologies (e.g. Prince2[8], PMBoK[9]), the quality of the development process is measured with reference to the quality of the products created thereby, including both final and intermediate products. Each of the products resulting from a development process should therefore be subject to specific and measurable standard and benchmark quality criteria, the compliance with which should be verified at the moment when the product in question is released. Products which do not comply with such criteria are either rejected or sent back for improvement.
The product Quality Management process (F.6.) should consist of two phases. During the first phase standards and quality requirements for the products are laid down, while during the second phase the products are subjected to a quality control check, according to the standards defined during phase one.

Phase 1. Quality requirement definition processes – process F.3-A (figure no. 3)

- **Preparatory activities preceding the process**

1. Pursuant to the agreement concluded with the client, a ***Product Catalogue*** *(PC*), a ***Product Succession Diagram*** *(PSD)* and ***Product Structure Diagram*** *(PSTD)* are prepared. These form the basis for the ***Project Timetable*.** The persons tasked with the preparation of the foregoing are most commonly the *Project Manager*, the *Quality Manager*, the *Systems Architect* (if such position exist) as well as *the Leaders of individual phases*.

- **Process activities.**

2. Being products, the Timetable, the PC and the PSD are subject to approval by the client and to registration in the ***Product Repository*** for the given project (the product library). The persons active in this phase are: the Client's Representative (for external approval) and the *Product Library Administrator* (for registration).
3. For each of the products, **Quality Requirements** are drawn up. These consist of both formal requirements (compliance with document templates) and substantive requirements (substantive evaluation of the given solution). Quality Requirements form the basis for both internal and external acceptance of the product during the quality control process (performed by the *Quality Manager*). Quality Requirements must be registered in the product library and are subject to internal acceptance by the *Project Manager*, as well as external acceptance by the *Client's Representative*.

The model for the quality requirement definition process is demonstrated by figure no. 3.

Phase 2. Quality Control – Process F.3-B (figure no. 4)

Each completed product must first be immediately *registered* in the product library, where it receives a "working version" status. Subsequently, the product is subjected to quality control by the

8 PRINCE2 (Projects In a Controlled Environment) - A project management process methodology
9 PMBoK (Project Management Body of Knowledge) is a set of standards and preferred solutions in the field of project management, compiled and published by members of the PMI (Project Management Institute).

Quality Manager. It is only completed products, and not the subsequent versions thereof, which are subject to the quality control procedure. The *Phase Leader* sends a notification (usually by e-mail) that the product is completed and ready for quality control. Quality control is effected by verifying the compliance of the product with the applicable quality requirements. As a result of the quality control procedure, the product may receive an internal approval or be sent back for improvements. In each case this means a change in status of the product (possible product status options: *working version*, *working version- returned for improvements*, *internal acceptance granted*, *submitted for external acceptance*, *approved by client*, *not applicable*). As a result of the quality control procedure certain remarks may appear. These are recorded directly in the product's documentation. In extraordinary instances where substantial changes were introduced or where the product suffers from critical errors, a ***Quality Control Protocol*** is prepared. The duration of the control procedure depends on the product and is set in cooperation with the *Project Manager* and the developer of the product. The quality control procedure is repeated until the product receives internal approval or until the time earmarked for its development, specified in the timetable, is exceeded. In the latter case the decision as to the future of the product in question belongs to the *Project Manager*.

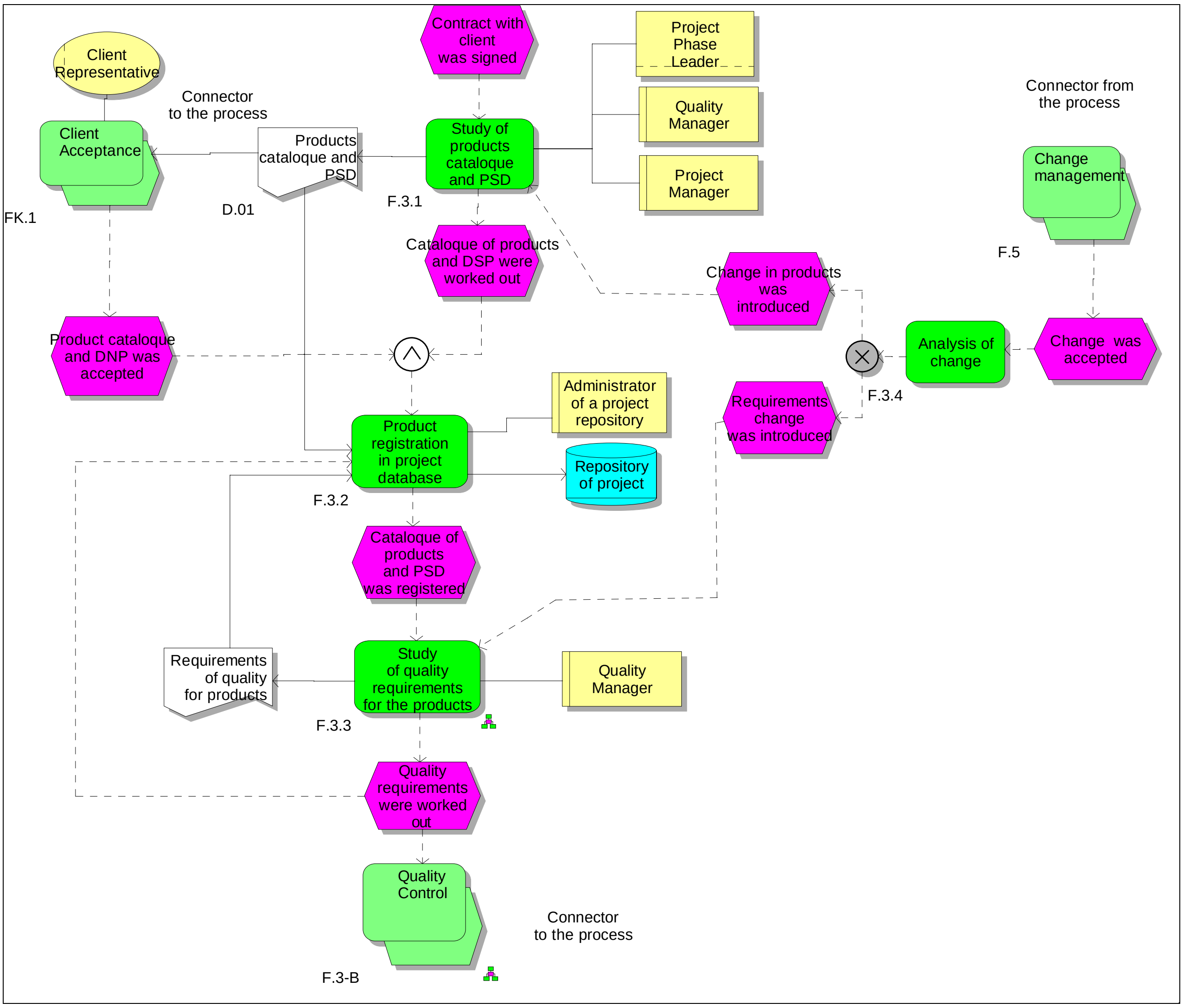


Figure no. 3. General referential model (eEPC- ARIS diagram) for the “Standard and Quality Requirement Definition” phase (Process F3-A). Source: Author's research.

Figure no. 4 demonstrates a sample model of the second phase: “Quality Control”, consistent with the above description. Both processes are closely interlinked by a connective process designated F.3-B “Quality Control”.

The ***Quality Control Process*** may interact with the ***Change Management Process***. Each approved ***Change Notification*** should be scrutinized by the Quality Manager as to whether it may require a quality control procedure to be performed or repeated for the given product, and also as to whether it may require the procedure itself to be amended (e.g. amendment of quality requirements, modification of the *Product Catalogue*, modification of the *Product Succession Diagram*, modification of the *Timetable*). In the first scenario (quality control), phase 2 is performed once again. In the second scenario stages 1, 2 or 3 of phase 1 are performed, insofar as necessary.

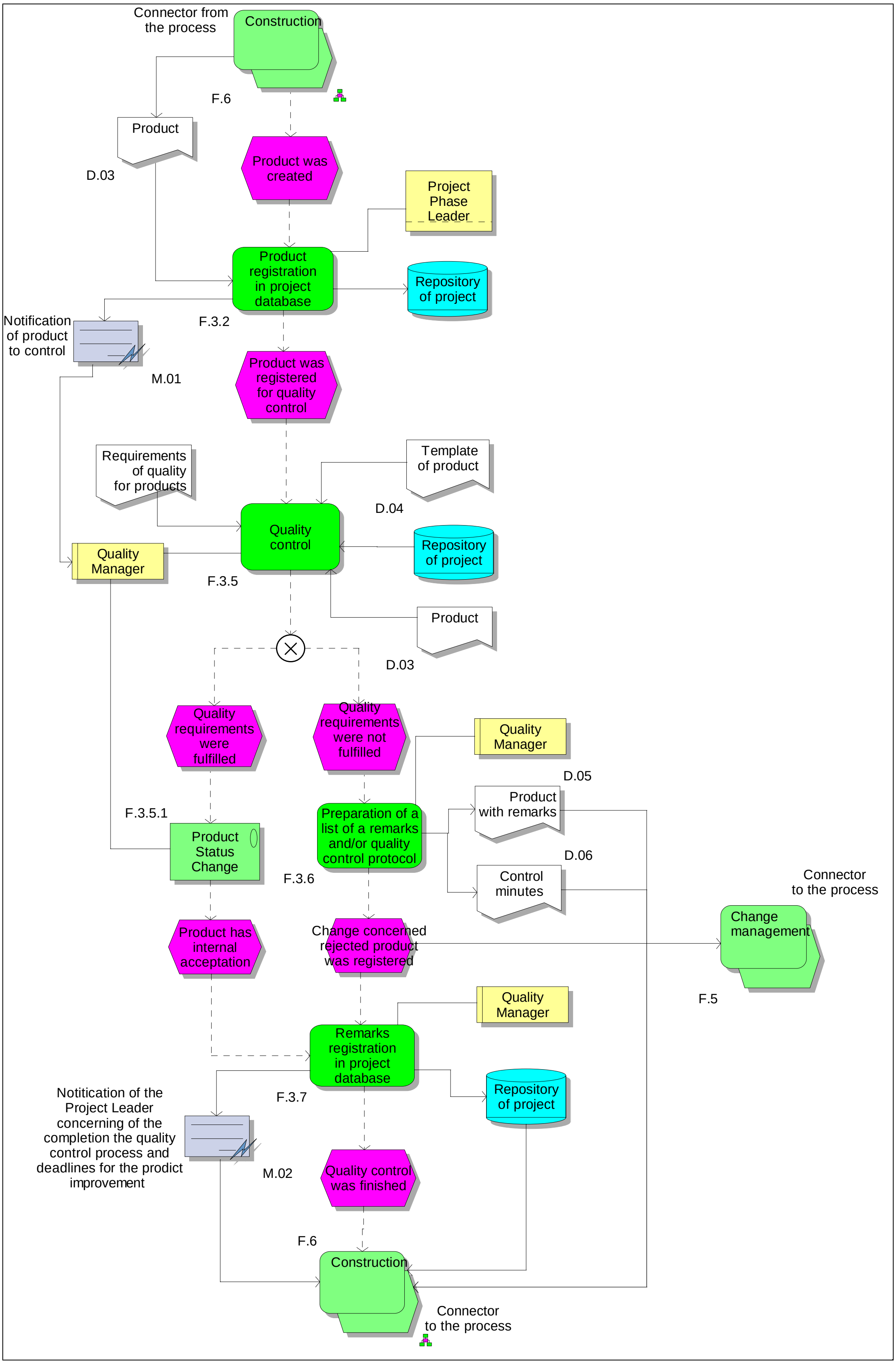


Figure no. 4. General referential model (eEPC- ARIS diagram) for the "Quality Control" phase (Process F3-B). Source: Author's research.

Table no. 3 shows a collation of processes which follow from figures 3 and 4 for both phases linked with the management of the quality control process.

Table no. 3. Quality management processes.

No.	Identifier on figures 3 and 4	Business function designation	Frequency
“Standard and Quality Requirements Definition” Processes (Figure no. 3)-Process F.3-A			
1.	F.3.1	Product catalogue and Product Succession Diagram (PSD) preparation	Once; optional amendment after change is introduced
2.	F.3.2	Product registration in the project database	Every time a product or a new version thereof is developed
3.	F.3.3	Product quality requirement definition	Once; optional amendment after change is introduced
4.	F.3.4	Change analysis	Each time a change is introduced
“Quality Control” Processes (Figure no. 4)- Process F.3-B			
5.	F.3.5	Product Quality Control	Every time a product or a new version thereof is developed
6.	F.3.5.1	Product status change	Optional- every time after quality control procedure is performed
7.	F.3.6	Compilation of remarks or quality control protocol	Every time after quality control procedure is performed
8.	F.3.7	Registration of changes in the project database	Every time after quality control procedure is performed; optional after change is introduced.

- **The “Use Case” model of quality management**

The quality management process can be demonstrated by means of a familiar UML “use case” model. It offers a different angle in the analysis of the issue which takes into account the point of view of the users (actors) of the quality management system. Figure no. 5 demonstrates this type of reference model.

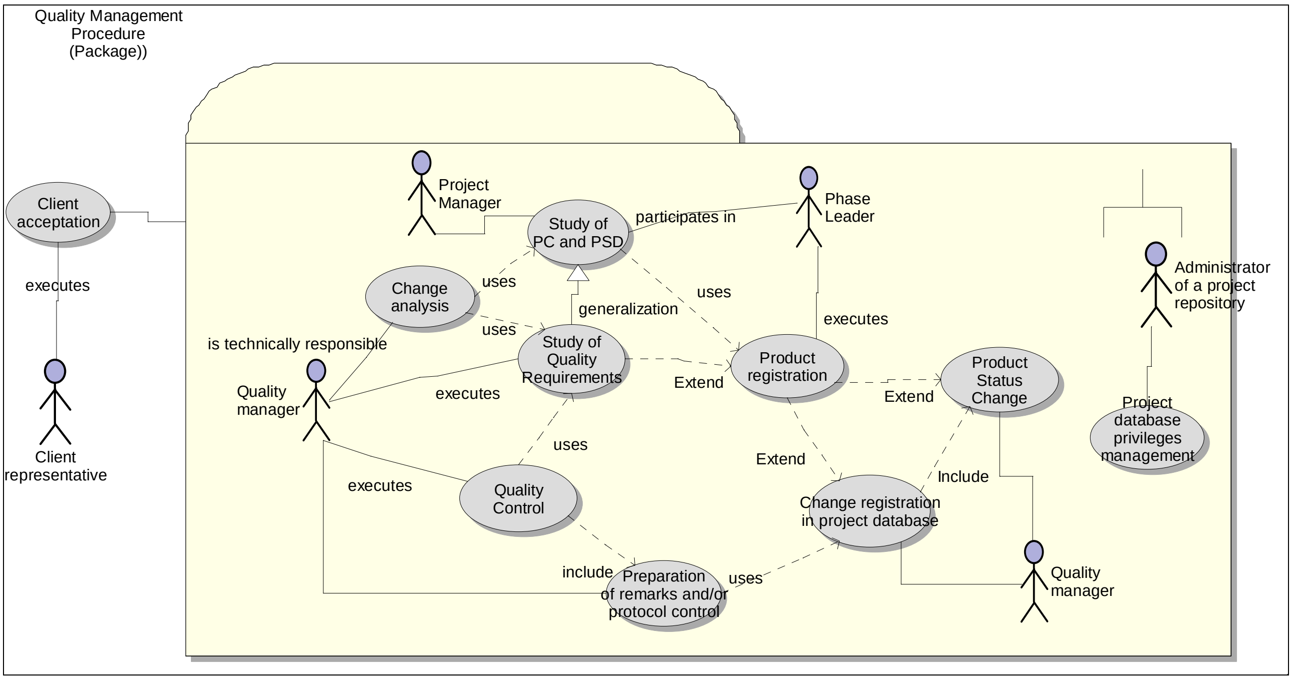


Figure no. 5: Referential model for “use cases for the Quality Management process”. Source: author's research.

5. Conclusions.

Quality management in the process of creating Business Intelligence-class systems is characterised by greater complexity when compared to ordinary code development, which is demonstrated by models included within this article- both those relating to the development process and to both phases of quality control (definition of quality requirements and standards and quality control). The present article attempts to systematise this issue by introducing certain reference models created by the authors. These models may form the basis for the creation of detailed models for specific applications and project solutions for BI systems. The reference models contained in this article ought to be viewed as a bridge between the realm of pure theory concerning software quality, specified and described in source [6], and specific solutions applied to real-life designs for this class of systems.

SOURCES

[1] A Guide to the Project Management Body of Knowledge. PMBOK Guide, third edition, Project Management Institute, 2004.

[2] DEO. N., Teoria grafów i jej zastosowania (The graph theory and its applications), PWN, 1980.

[3] INMON W. H.: Building the Data Warehouse. John Wiley & Sons, Inc., 1996.

[4] JACOBSON I., Object Oriented Software Engineering, Addison-Wesley Publishing Company, 1995, ISBN: 0-201-54435-0.

[5] KIMBALL R., The Data Warehouse Toolkit. John Wiley & Sons, Inc., 1996.

[6] KOBYLIŃSKI A. , Modele jakości produktów programowych i procesów programowych, praca habilitacyjna, monografie SGH, Warszawa 2005 (Software product and process quality models- habilitation thesis, SGH monographs, Warsaw 2005).

[7] Managing Successful Projects with PRINCE2, Crown 2006, ISBN 0 11 330946 5.

[8] ROSZKOWSKI J., Analiza i projektowanie strukturalne (Structural analysis and design), Helion, 2004, ISBN: 83-7361-397-8

[9] ROSZKOWSKI J., The Formal Approach to Definition of the Requirements for the Needs of Computer Application Selection and Implementation, Proc. of the 16th International Conference on Systems Science ICSS'07, 2007
[10] SCHEER A. W.: Business Process Modeling, Springer-Verlag, Berlin, Heidelberg ,New York, 2000, ISBN: 3-540-64438-5.
[11] SCHEER A.W., Business Process Engineering, Springer-Verlag, Berlin, Heidelberg, New York, 1994, ISBN: 3-540-51480-5.
.