A Data Warehousing Environment to Monitor Metrics in Software Development Processes

Duncan D.A. Ruiz, Karin Becker, Taisa C. Novello, Virginia S. Cunha
Faculdade de Informática
Pontifícia Universidade do Rio Grande do Sul – PUCRS
{duncan, kbecker, tnovello, vcunha}@inf.pucrs.br

Abstract

Software organizations follow capability models in order to aggregate quality to their processes and products. Process measurement and analysis are key issues, but their implementation must consider that there is an enormous diversity in how projects are developed, even in the same organization. This work introduces a Data Warehousing environment to support the implementation of a measurement program in an organization currently certified as CMM Level 2. The environment addresses in an integrated manner three key issues: 1) data capturing, considering various types of heterogeneity; 2) the integration, transformation and representation of project quantitative data according to a unified and centralized organizational view; and 3) analytical functionality allowing process monitoring.

1. Introduction

Software organizations are facing growing demands for better quality software, shorter development time and lower costs. Software capacity models (e.g. CMM and CMMI[1]) guide organizations to identify best practices to increase the maturity of their Software Development Processes (SDP). SDP quality may be quantified through a measurement program designed to monitor, detect and prevent flaws. Organizations are seriously concerned with the definition of metrics representative of the target quality areas. However, the definition of metrics alone is insufficient. A big challenge is to collect and represent heterogeneous data relative to different projects, according to a unified organizational view. An organizational database that integrates data of different projects is one of the requirements to evolve in the maturity levels of CMM and CMMI.

[1] http://www.sei.cmu.edu/cmm/cmms/cmms.html

However, the design and development of such a repository must address many challenges. Projects in the same software organization may differ in terms of development processes, tools adopted, as well as strategies employed to generate, store and control project-related data. In addition, organizations differ in terms of the OSSP (Organization's Standard Set of Processes) adopted, according to their maturity level. Presently, there is no generic support infrastructure for a measurement repository that takes into account all the above-mentioned differences. Specific proposals can be found in [1][2][3][4][5].

Data Warehouse (DW) is an integrated, non-volatile, historical, subject-oriented data collection, aimed at supporting decision-making processes [6]. Data Warehousing is the process for assembling and managing the DW, encompassing business modeling; data extraction, transformation and loading (ETL); querying and analytic tools targeted at end-users; and repository management and maintenance functionality.

This paper introduces a Data Warehousing environment to support the adoption of measurement program in a large software organization. The organization is currently certified as CMM Level 2. The core of the environment is the DW, in which project-related data is stored to support the monitoring of SDP according to the defined metrics. The goal is to provide a centralized and unified view of all projects, together with functionality that allows simple and straightforward analyses on SDP metrics, according to different analysis perspectives, summarization levels, and organizational roles. The architecture also encompasses a non-intrusive approach for the capturing of project-related data from operational data sources, which addresses various types of heterogeneity.

The remainder of this paper is structured as follows: Section 2 discusses quality in SDP; Section 3 details the Data Warehousing environment proposed to support SDP measurement and analysis; Section 4 discusses related work; Section 5 presents conclusions.

2. Quality in SDP

2.1. Metrics

Metrics are essential in the assessment of software development quality. They may provide information about the development process itself and the yielded products. Metrics may be grouped into Quality Areas (QA), which define a perspective for metrics interpretation. The adoption of a measurement program includes the definition of metrics that generate useful information. To do so, organization's goals have to be defined and analyzed, along with what the metrics are expected to deliver.

Metrics may be classified as direct and indirect [7]. A direct metric is independent of the measurement of any other. Indirect metrics, also referred to as derived metrics, represent functions upon other metrics, direct or derived. Productivity (code size/ programming time) is an example of derived metric. [7] Presents a critical discussion on the difficulty in establishing a valid measurement program, as well as a framework to assess and understand metrics. The existence of a timely and accurate capturing mechanism for direct metrics is critical in order to produce reliable results.

Indicators establish the quality factors defined in a measurement program [7]. The quantification of indicators according to organization's metrics produces information that enables to evaluate the quality of SDP. Consider for instance the indicator *defect density* (low, medium, high). The intended quality factor may be *low defect density* (e.g. less than 0.3 defects/KLOC).

2.2. Measurement and Analysis Requirements

CMM and CMMI aim to provide organizations with a shared view of their SDP performance. In CMMI, Measurement and Analysis (MA) is a process area that aims at developing and maintaining a measurement capacity that meets management information needs. MA influences all other process areas, and recommends the storage of project-related data in a repository. The more repository data is shared between projects, the more comprehensive and effective is the organization's ability to assess its PDS.

CMMI-Level 2 states that an organization has to define metrics aligned with its goals and information requirements. These metrics have impact on the possible analyses and on future maturity levels. Level 3 states the requirement of an organization repository that establishes a unified view of its processes. This repository has to (i) store products and process metrics previously defined by the organization in the OSSP and (ii) handle information enabling quantitative process evaluation. The metrics defined in Level 2 and the organizational repository of Level 3 support better project estimates and planning. CMM requirements are very similar to the ones of CMMI.

CMMI-Level 4 aims to establish and maintain the quantitative understanding of the organization's process and baselines, in addition to make quantitative management models available. Level 5 is targeted at SDP optimization. These two levels are also based on the availability of an organizational repository, which provides enhanced analytical functionality.

3. A SDP Data Warehousing Environment

This section presents a Data Warehousing environment focused on SDP metrics for the target software organization, which aims to be certified as CMM Level 3 in a near future. According to [7], MA requires that the organization define which metrics are to be adopted, the respective measurement units, and the manner in which these metrics can be put together to form other metrics. Table 1 shows the set of metrics considered by the target organization and the quality areas to which they refer.

Table 1: Metrics Adopted.

QA	Derived Metric	Direct Metric
Time	Schedule Variance (Original and Revised Baselines)	ASD – Actual Start Date AED – Actual End Date OBSD – Original Baseline Start Date OBED – Original Baseline End Date RBSD – Revised Baseline Start Date RBED – Revised Baseline End Date
Work	Effort Variance (Original and Revised baselines)	AE – Actual Effort OBE – Original Baseline Effort RBE – Revised Baseline Effort
Size	Size Variance (Original and Revised baselines)	OBES – Original Baseline Estimated Size RBES – Revised Baseline Estimated Size AS - Actual size
Cost	CV - Cost Variance SV - Schedule Variance CPI - Cost Performance Index CSI - Scheduled Performance Index	BCWS – Budgeted Cost of Work Scheduled BCWP – Budgeted Cost of Work Performed ACWP – Actual Cost of Work Performed
Defect	Defect Removal Efficiency Defect Density Review Efficiency	NDIF – Number of Defects Internally Found NDFC – Number of Defects Found by Clients

The proposed architecture is composed of 3 layers: application integration, data integration, and presentation (Figure 1). The application integration layer extracts project-related data from operational sources. The data integration layer comprises the DSA (Data Staging Area) and the DW itself. DSA is a temporary database into which raw data is transferred, cleaned and transformed, prior to DW loading. ETL acts on the application integration and data integration layers, and follows a service-oriented approach [8]. The presentation layer enables metric-oriented SDP monitoring and analysis. These aspects are discussed in more detail the remainder of this section.

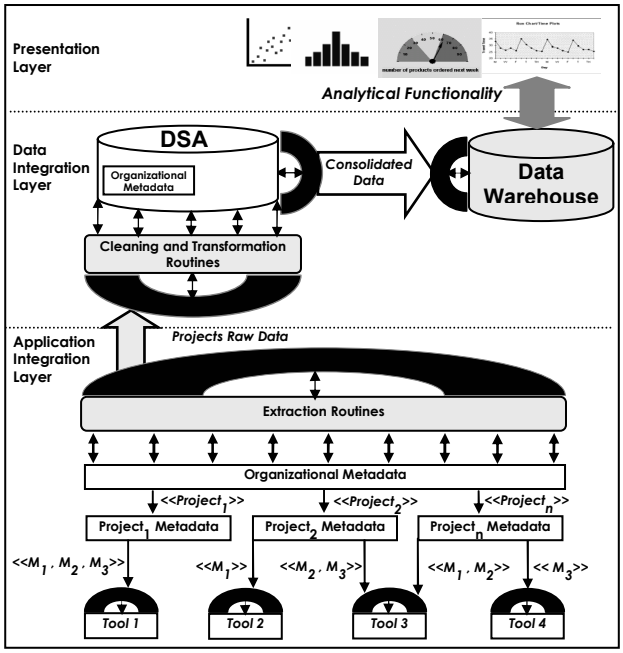

Figure 1: Data Warehousing Architecture.

3.1. The Data Warehouse

The DW is the unified and centralized database that supports the organization's metrics program. The analytical must enable analyses on SDP according to different perspectives, summarization levels and organizational roles. Its design was guided by the structure of software projects in the target organization, which is depicted in Figure 2, using a UML class diagram. Attributes in capital letters refer to Direct Metrics of Table 1. Projects are organized in phases, and yield products. Cost, schedule and effort estimates are established in terms of phases, whereas size estimates are established in terms of products. The actual measurements for these metrics are captured as phases are developed through activities, and yield products. Defects are measured as products evolve through phases.

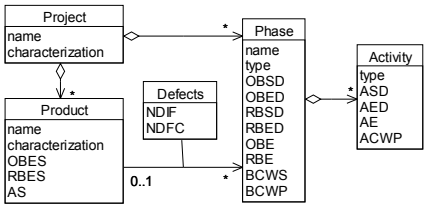

Figure 2 – Project Structure and Direct Metrics

Multidimensional modeling is widely used to represent data in the DW [6]. It is a model suitable for establishing analysis according to multiple perspectives and summarization levels, and it is more intuitive than "flat" (normalized) models.

The proposed analytical model, depicted in Figure 3, is a constellation of facts. Dimension tables, which represent project analysis perspectives, are described in Table 2. Fact tables store direct metrics at different granularity levels. For instance, actual work can be analyzed at activity level, whereas size is a coarser-grained measure, which is established for the product as a whole. These decisions were guided by the availability of data, as represented in Figure 2. Derived metrics are not stored, given that hey are not additive. Table 3 briefly describes fact tables and relates them with QA of Table 1. It is important to highlight that estimated and actual measurements are recorded as distinct facts the same table, and are distinguished by the dimension to which they are related (Type_Fact_Dim, in Table 2).

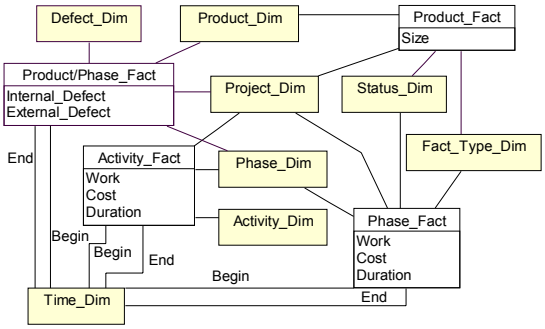

Figure 3: The Analytical Model.

Table 2: Dimensions of the Analytical Model.

Dimension	Description
Project_Dim	Data characterizing projects. (e.g. category, technology, client, etc)
Product_Dim	Data characterizing software products
Phase_Dim	Data characterizing project phases.
Activity_ Dim	Data characterizing Project phase activities, classified according to they type (work, rework, revision)
Defect_Dim	Data characterizing defects, based on category (intern or external) and severity (low, medium or high).
Status_Dim	Status (on-going or completed) of a product or phase of a project.
Time_Dim	Date description (date, year, month, day and semester).
Fact_Type_Dim	Type of fact (original baseline estimate, revised baseline estimate or actual value).

Table 3: Facts of the Analytical Model.

Fact	Description	Related QA
Activity_Fact	Metrics quantifying an activity of a project phase (actual values)	Time Work Cost
Phase_Fact	Metrics quantifying estimates and actual values related to a project phase	Time Work Cost
Product/Phase_Fact	Metrics quantifying estimates and actual values related to a product in a specific phase of the project	Defect
Product_Fact	Metrics quantifying estimates and actual values related to a specific product	Size

3.2. Service-Oriented ETL Model

ETL is a complex process, of which the ultimate goal is to provide DW users with relevant, concise and quality data that supports business decision-making. In the SDP context, ETL aims to capture data on the execution of processes, as well as on the artifacts produced and handled by them. ETL must consider project idiosyncrasies, which, from a data capturing perspective, implies dealing with different types of heterogeneity: different tools, diverse approaches for recording project data (even when the same tool is used), and specializations of OSSP processes. Another issue is that each tool has a proprietary data model. ETL covers both the application integration and the data integration layers of the data warehousing architecture.

In the target software organization, projects have the freedom to choose the tools considered appropriate for project planning, execution and management. Such tools range from conventional spreadsheets (e.g. MS Excel) to dedicated project supporting tools (e.g. MS Project Server, IBM Rational ClearQuest), and store data in different places and formats. Projects may adopt specialized processes from the OSSP, introducing variants in the lifecycle and management model. Hence, process data is recorded differently even when the same tool is adopted. Additionally, most tools have limited functionality for data extraction, requiring *ad-hoc*, laborious and error-prone extraction procedures.

The application integration layer is responsible for extracting project raw data from diverse tools and loading it in the DSA. We adopted a low-intrusion approach by the use of wrappers, according to a service-oriented architecture. Every wrapper addresses the extraction of data considering a particular tool, and its underlying proprietary data model. In addition, each project is described by metadata, expressed as XML schemata. Project metadata defines the tools adopted, and how required data (i.e. metrics and dimensions attributes) are recorded in these tools according to the project life cycle (e.g. iterative, cascade), type (e.g. development, maintenance) and management model (e.g. delivery-oriented, phase-oriented). Extraction procedures exploit project metadata to locate the correct wrapper, and to guide the extraction based on the mapping established between the raw data and the required data. This approach has the following advantages. First, it allows reducing implementation complexity and maintenance of extraction procedures, given that it separates project description from the extraction procedures. Additionally, projects may evolve over time (e.g. adoption of new tools, change of management model). It enables the handling of heterogeneity with the use of standard protocols (XML, SOAP, WDSL and UDDI) [8]. These protocols are used to define message formats, specify the destination interfaces to deliver messages, describe the transformation rules to map incoming and outgoing message contents, and to publish and find services.

In the data integration layer, data in the DSA is cleaned and transformed, and subsequently loaded into the DW. Metadata supporting this layer provides a unified view of all projects from the organization perspective, and their mapping in the DW analytical model. For example, project *A* must classify the severity of defects as {1, 2, 3, 4}, whereas project *B* uses {A, B, C}. Metadata establishes transformation rules that convert these project-specific scales into the organizational scale (e.g. {High, Medium, Low}).

3.3. Presentation Layer

The presentation layer of a data warehousing environment grants users access to DW data according to their different profiles and objectives [6]. Executives, software quality assurance team and project team may require management and decision making information according different perspectives (dimensions) and abstraction levels (in more or less detailed ways). The presentation layer should offer query and visualization tools with different degrees of sophistication, standardized reporting tools and specific-purpose applications. OLAP (On-Line Analytical Processing) provide operations that explore the multidimensional model as a cube (slice and dice, drill down/up). Specific applications enable pre-defined, parameter-based analyses of specific organization sectors. The analyses may be presented using different types of charts, tables and reports. Each type of tool adjusts to the needs of a given user profile, considering the level of activities performed (i.e. strategic, managerial, operational), and technical skills.

The analytical model proposed enables a variety of queries over the metrics, by navigating through the dimensions. All derived metrics can be obtained by queries, at different levels of granularity. For instance, consider the metric original baseline schedule variance = (AED - ASD)/ (OBED - OBSD). It can be calculated for each phase, for a project as a whole, it can be constrained by specific properties of projects/phases, etc. The architecture provides reliable analysis approaches to end-users by the use of tools that combine OLAP operations, available through different presentations (predefined reports, parameterized queries, graphics, control board, etc). Figure 1 sketches some examples. In addition, our approach provides users with facilities concerning quality indicators. Indicators enable the monitoring of organizational processes executions, alerting users every time an unexpected behavior is detected.

4. Related Work

The database discussed in [1] includes a limited set of metrics, captured by project management dedicated tools. The approach establishes constraints on projects, and do not support extensions easily. Multidimensional Metrics Repository (MMR) [2] is a flexible metrics repository, which exploits OLAP for SDP monitoring. Although two data loading processes (manual and semi-automatic) are mentioned, no further ETL issue is discussed. Our DW design is not as flexible as MMR, but the warehousing approach include a complex ETL encompassing various types of heterogeneity.

Business Process Intelligence (BPI) [4][5] is a general-purpose business process monitoring data warehouse environment. BPI architecture considers a broad number of issues, ranging from ETL to enhanced functionality for data presentation. Business Cockpit is the interface component focused on the follow-up of processes via metrics and indicators. However, BPI does not take into account SDP specificities, such as the different types of heterogeneity discussed, and SDP metrics that are essential to the maturity certification processes. [3] Presents a framework that combines service-oriented architectural concepts, principles of decision-support systems, and a multi-agent approach to analyze business process status and performance. As our approach, it deals with various types of heterogeneity in business processes by using a service-oriented architecture. Its striking contribution is the efficient and timely capturing of data. Analyzes over data are also implemented as dedicated web services, providing less flexibility for end users in the spectrum of possible analyses and alternative presentations, if compared to a data warehousing environment.

5. Conclusions

This paper proposes a Data Warehousing environment as supporting infrastructure for a MA program of a software organization aiming to be certified as CMM level 3. It integrates three essential aspects: 1) data capturing, considering various types of heterogeneity; 2) the integration, transformation and representation of project quantitative data according to a unified and centralized organizational view; and 3) analytical functionality allowing process monitoring. An experiment that validates our architecture using actual organizational data has been designed. An environment prototype is currently under development, using SQL Server and associate OLAP tools, .Net and which explores various types of heterogeneities involved in the use of MS Project.

The analytical database was designed taking into account a restricted but significant set of metrics. New metrics may be easily included, provided that they are compatible with the way process and products are represented through the conformed dimensions [6]. Metrics of larger or smaller granularity may be stored by creating new fact tables.

The service-oriented ETL approach has the following advantages: 1) to deal with several types of heterogeneity with services that act as wrappers of the different types of extractors and project metadata that parameterize extraction procedures, 2) to enable a non-intrusive mechanism to extract the measurements, and 3) to support a distributed development environments. The first two advantages are essential for the acceptance of a metric program. It imposes fewer constraints on projects, allow the adoption of new tools, and increase the accuracy of collected data by eliminating manual procedures.

Future work includes (1) the development of more advanced analytical resources (e.g. mining of historical data), (2) the refinement of the architecture proposed to implement an entirely service-oriented architecture, and (3) an empirical evaluation of the use of the environment inside the organization.

Acknowledgements

This work is supported by HP Brasil Ltda., under HP Brasil/PUCRS agreement TA 02 HP 001/03.

References

[1] V. Subramanyam, S.V.B. Sharma, “HPD - Query tool on Projects Historical Database”, SEPG Conference in Bangalore, Feb 1999.

[2] E. Palza, C. Fuhrman, A. Abran. “Establishing a Generic and Multidimensional Measurement Repository in CMMI context”. *28th Annual NASA Soft. Eng. Workshop (SEW’03),* pp.12-20. IEEE, *Dec. 2003*.

[3] C. McGregor, J.A. Schiefer. “A Framework for Analyzing and Measuring Business Performance with Web Services”. IEEE Intl. Conf. E-Commerce (CEC’03), pp.405-412. IEEE, June 2003.

[4] D. Grigori, F. Casati, M. Castellanos, U. Dayal, M. Sayal, M. Shan. “Business Process Intelligence”. *Computers in Industry*, v.53, n.3, pp.321-43, Apr 2004.

[5] M. Castellanos, F. Casati, U. Dayal, M. Shan. “A comprehensive and automated approach to intelligent business processes execution analysis” *Distributed and Parallel Databases*. v.16, n.3, pp.239-73 Nov. 2004.

[6] R. Kimball. *Data Warehouse Toolkit*. São Paulo: Makron Books, 1998.

[7] C. Kaner, W.P. Bond, “Software Engineering Metrics: What do they measure and how do we know?” *10th Intl. Software. Metrics Symposium,* Sept, 2004.

[8] G. Alonso, F. Casati, H. Kuno, V. Machiraju, *Web Services – Concepts, Architectures and Applications*. Springer, 2004.