ÉCOLE
POLYTECHNIQUE
MONTRÉAL

Le génie sans frontières

Time Monitoring Tool
Measurement Plan

Version 2.0

Revision History

Date	Version	Description	Author
<22/01/01>	1.0	First metrics entries	John Lemon
	1.1	Developed metrics	John Lemon
	1.2	Reviewed and added metrics	John Lemon
	2.0	Page layout and global review (corrections)	John Lemon

Preface

The following case study has been modified from its original content.

The case study is meant to be used as a starting point to help you understand how to use the artifact. Thus, information has been shrunk to avoid navigating an enormous document (in size and pages). Note that The Measurement Plan may physically be part of the Software Development Plan if the metrics program is a simple one.

You can also refer to the related template (in HTML format or WORD format) in the UPEDU Artifacts Templates Project Management Section.

Regards,

Unified Process for Education Team

Table of Contents

Measurement Plan

1. Introduction

1.1 Purpose

The purpose of the Software Measurement Plan is to specify the primitive metrics which should be collected and the metrics which should be computed during the project. For the project progress monitoring, a set of project goals should be specified.

1.2 Scope

This Software Measurement Plan defines a simple metrics program, including the measurement goals, the associated metrics, and the primitive metrics to be collected in the project, in order to monitor its progress. The details of the individual activities will be described in the Iteration Plans. The plans as outlined in this document are based upon the software requirements as defined in the SRS Document.

1.3 Definitions, Acronyms, and Abbreviations

See the Project Glossary

1.4 References

For the Software Measurement Plan, the list of referenced artifacts includes:

- **SRS**
- **Software Development Plan**
- **Iteration Plans**
- **Project Measurements**
- **Glossary**

1.5 Overview

This *Software Measurement Plan* contains the following information:

Measurement Goals: **The goals of the measurement program relative to the project in terms of achievement, improvement, and quality.**

Metrics: **The metrics that are to be synthesized at regular intervals on the project to support the goals**

Primitive metrics: **The primitive metrics that are collected, automatically or manually, to compute the metrics**

2. Measurement Program Goals

There are essentially two approaches that can be followed to ensure end product quality, one being assurance of the process by which a product is developed, and the other being the evaluation of the quality of the product.

- **the Process — the sequence of activities invoked to produce the software product (and other artifacts)**
- **the Product — the artifacts of the process, including software, documents and models**

The goals of the Measurement Program relative to the Time Monitoring Tool project are the following:

- **Evaluate:**
  - **the quality of the product**
  - **The quality of code**
  - **The quality of documents and models**
- **Assure**
  - **a documented follow-up of the project**
  - **the progress of tasks according to estimates**
  - **the state of the activities**
- **Evaluate**
  - **the impact of the changes to the requirements**
  - **tasks implied by the requirements**
  - **The realization level of the requirements**
- **Facilitate**
  - **the planning and the estimate of new projects**
  - **The relative importance of disciplines**
  - **The importance of the activities inside disciplines**

3. Metrics

3.1 Product

The products are the code, the documents and the models. Some specific characteristics and their associated metrics are listed in the following table:

	Characteristics	Metrics
Code	Size	SLOC
	Size under configuration control	SLOCc
	Depth of inheritance tree	DIT
	Adaptability	E/N, average effort per SCO
Documents	Size	Page count
	Effort	Staff-time units for production, change and repair
	Volatility	Number of changed pages
	Traceability	Not measured directly: judgment made through review, using a checklist
Models	Size	Number of Use Cases Number of actors Number of classes Number of packages Methods per class Attributes per class Number of Test Cases

3.2 Process

To completely characterize the process, measurements should be made at the lowest level of formally planned activity. Activities will be planned by the Project Manager. A record should then be kept of actual values over time and any updated estimates that are made.

Metrics	Comments
Effort	Staff effort units (staff-hours)
Duration	Elapsed time for the activity
Output	Artifacts and their size and quantity (note this will include defects as an output of test activities)
Effort application	The way effort is spent during the performance of the planned activities

3.3 Metrics

Name	WMC
Definition	The Weighted Methods per Class (WMC) is a count of the methods implemented within a class.
Goals	The number of methods involved is a predictor of how much time and effort is required to develop and maintain the class. The larger the number of methods in a class, the greater the potential impact on children since children will inherit all the methods defined in a class. Classes with large numbers of methods are likely to be more application specific, limiting the possibility of reuse. This metric measures usability and reusability.
Analysis Procedure	The Weighted Methods per Class will be counted by the software design tool – Rational Rose. The data will be collected al the end of the corresponding activities for the Design Class Model artifact. The data will be saved in the Measurement Data Base.
Responsibilities	Software Engineering Team

Name	Duration
Definition	The duration will be calculated based on start and end time: Duration = End_Time – Start_Time The Duration will be calculated in minutes. The Start and the End Time were approximated to the nearest quarter of an hour.
Goals	The Duration metric is being collected to permit the effort calculation. Effort measurement will be used for product measurement and for process measurement.
Analysis Procedure	All team members will use the same template for the effort recording information, presented in Annex / point 5.1. The tool used to register the time spent on each activity is presented in Annex / point 5.2. The data will be saved in the Measurement Data Base.
Responsibilities	Each team member will be responsible for filling out his records capturing the time spent on a specific activity, once the activity is ended. One team member will be responsible for collecting every week his team-mates' completed records and for integrating the data into the Measurement Database.

4. Primitive Metrics

The following metrics will be collected manually to compute the metrics:

Name	SLOC
Definition	Source Line of Code
Collection procedure	SLOC will be count by the programming environment. The data will be collected at the end of each iteration. The data will be saved in the Measurement Data Base.
Responsibilities	Software Engineering Team

Name	SLOCc
Definition	Source Line of Code for the current baseline
Collection procedure	SLOC will be counted by the programming environment. The data will be collected each time the base line is changed. The data will be saved in the Measurement Data Base.
Responsibilities	Software Engineering Team

Name	Number of Use Cases
Definition	Number of Use Cases in the Use Case Model
Collection procedure	The use cases will be counted by the software design tool – Rational Rose. The data will be collected al the end of the corresponding activities for the Use Case Model artifact. The data will be saved in the Measurement Data Base.
Responsibilities	Software Engineering Team

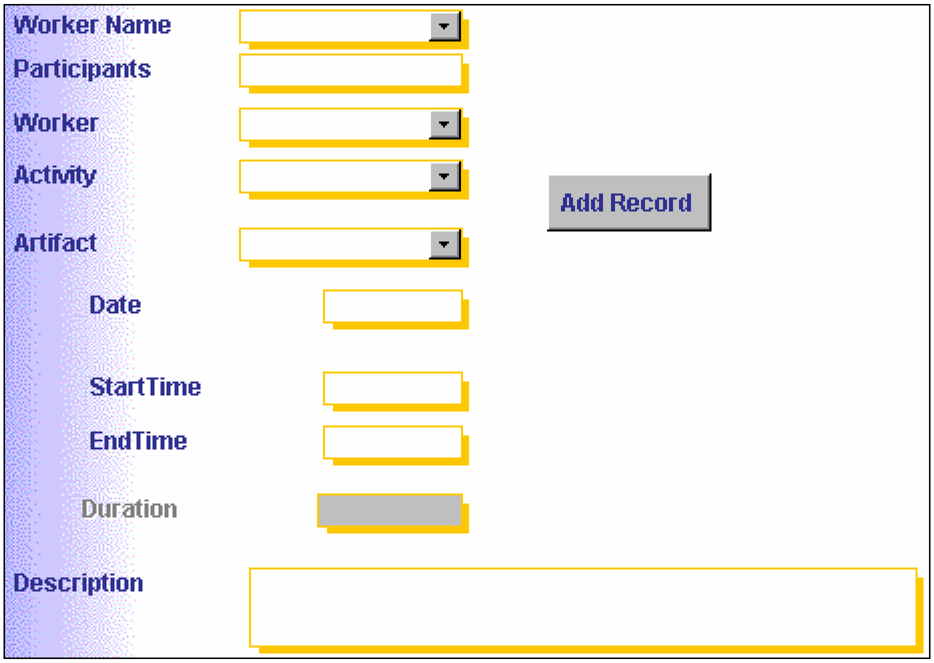
5. Annexes

5.1 Format of effort recording information

Worker Name	Worker	Activity	Artifact	Date	Start Time	End Time	Duration

For each record, the students inserted a short description of the entered data.

5.2 Tool used to register the time


Worker Name
Participants
Worker
Activity
Artifact
Date
StartTime
EndTime
Duration
Description
Add Record