Time Metrics Using the Shlaer-Mellor Method

"How long is this going to take?"

Rodney Montrose

Project Technology, Inc.
10940 Bigge Street
Sam Leandro, CA 94577-1123
(510) 567-0255
http://www.projtech.com

When an organization begins using a new software development process, its management often wonders about the risk and the effects this new approach will have on the group. Effects on productivity, quality of work, job satisfaction, and time to market are issues of concern. When a project starts with a new development process, managers often wonder if too much time is spent on the various activities and when the project will be completed. Since they do not have prior experience with this new development process, there is no standard to judge against. This is especially true if the development is done using the Shlaer-Mellor Method, which prescribes lots of up front work (analysis) and postpones traditional programming efforts (coding) until much later in the software development process.

The Shlaer-Mellor Method[1] defines more than just a notation and set of models for Object-Oriented Analysis (OOA) and design; it prescribes a repeatable, defined process for software development. The steps and the activities performed in each of these steps influence how the time is measured in each step. To understand the expected amount of time something needs to be accomplished, you must first be able to measure the current activities being performed. The activity of performing these measurements is commonly referred to as "collecting metrics." A metric is defined as "A standard of measurement." [2]

This paper will describe the application of time metrics to developers using the Shlaer-Mellor Method. It will cover what should be measured, how to conduct these measurements, and how the Shlaer-Mellor Method influences these numbers. The paper also includes some sample results we have collected. These are intended to provide a framework so that an organization can implement their own measurement program. They are not intended to provide a set of numbers carved in stone that each group should be measured against. Project Technology has seen that the time numbers vary among groups and projects; however, within a specific project the time measures are consistent for each step of the method.

It is assumed that the reader understands the Shlaer-Mellor Method,[1,3] especially the concept implementation through translation.[4] Because of the uniqueness of this approach, the use of these metrics for other OO methods will be limited.

Measurement: The Shlaer-Mellor Method can be characterized as being built up of many well-defined steps. Each step is consistently applied to the problem in a manner defined by the method, not in an inconsistent manner defined by the individual analyst. This feature allows many opportunities to measure the time taken in each of the steps. With a sufficient sample size, repeatable numbers can be derived and applied against new measurements. Some companies using the Shlaer-Mellor Method have developed and recorded detailed metrics expected for each step of the process.[5]

There can be various granularity levels in the time measurements taken. Some examples of these measurements are:

High level metrics
- Time per Domain
- Time per Subsystem
- Time for each Information and State model
- Time for each Action Specification

Middle level metrics
- Time per Object (all steps of OOA)
- Time per Object (from Object Information Model (OIM) through test)
- Time for each step of OOA (OIM, State Models (SM) or Action Specification (AS))

Low level metrics
- Time per State or Action Specification
- Time to write an Object or Relationship description

Higher level metrics will present more accurate and repeatable measurements since they cover a longer period of work and are less likely to be affected by individual analyst performance. The finer the level of resolution, the greater the effect from individual analysts and the less repeatable are the measurements.

To perform these measurements, accurate time records must be religiously maintained. We have seen the best resolution that can be accurately maintained by individual humans is half hour increments. Some companies use special watches or computer programs that alert the analyst to record what they are doing at that particular moment. These types of measurements are as accurate as the measurements where each activity is individually recorded ("We just spent 1 1/2 hours on this state model."). Regardless of the actual method of measurement, accurate results are achieved only when the information is recorded as the activity is being performed. Recording this information at the end of the week just doesn't work; people cannot accurately determine how their time was spent that long ago.

Collection of these metrics results in large amounts of data that must be collated, reduced and maintained. Project Technology has seen groups using special databases, spreadsheets and paper to capture this information. Many of the results from these measurements can be recorded into a Project Matrix[6] against the step in the process they are measuring.

Real Project Time: It is also important to realize that no one can perform a single activity continuously. Interruptions are a normal part of most work environments. This has two effects: fractured time and non-productive time. Fractured time is when a single task is spread over many discreet time intervals. As the number of intervals increases, more time is spent getting back to a productive phase. Our measurements must record the sum of all the fractured time elements. Nonproductive time is time spent doing things other than analysis activities. This time shouldn't be recorded as analysis time, otherwise inflated times will be recorded.

It is also important to communicate to the developers that the time measures are not done to measure an individual's productivity, otherwise the numbers collected will be erroneous since they will be biased to show that the person is very productive. Accurate metrics are collected only when the measurement process doesn't affect the time being measured. This can potentially be a major pitfall in instituting a metrics program, and it requires the full support of management to eliminate this potential problem.

When you are measuring the time, it is important to distinguish between the analysis, review and correction time. It is important to also separate the administrative time (training, vacation, illness, other project activities, and other miscellaneous items that arise during a project) from the actual time spent on the project. Having 20% or more of a project's time spent on administrative activities is not uncommon.

Dave Lewinski[7] defined a metric called the Applied Effort Factor (AEF) which measures the effort applied towards the project compared to the overall effort. It is defined as:

$$AEF = \frac{\text{Project Time}}{(\text{Project Time} + \text{Non-Project Time})}$$

If the AEF on a project drops to 60% or less, an investigation should be carried out as to the reason for this large expenditure of time not on the project.

Process Improvement: The purpose of time measurements is to 1) see how long each activity takes and shorten the cycle time, 2) to minimize the variations in each step. During the first use of the Shlaer-Mellor Method, it is expected that following the steps in the process will take longer than when experience has been reached. Some suggestions to improve the time taken in the various steps are:

Object Information Modeling (OIM) - Are the requirements specified and completed before the analysis begins? Have the domains been properly identified and separated? Are the people finding the objects expected in this domain? Do the analysts have access to domain experts? If any of these are not in place, they can significantly increase the time to perform the object analysis and build the OIM.

State Modeling (SM) - Have the objects in the IM been abstracted correctly? Is the behavior well understood? Have communication patterns been established? Again, if these haven't been done the state modeling will be longer than expected.

Action Specification (AS) - Have the data query paths been diagrammed? Are the tester and transformation processes understood?

Recursive Design (RD) - Start by examining the time to translate the OOA models into implementation code. Automation through a new generation of Computer Aided Software Engineering (CASE) tools[8] can reduce this time from 30% to 10% or less of the total project time. One of the principle strengths of the Shlaer-Mellor Method is the ability to automate the production of 100% of the implementation from the analysis (OOA) models.

Metrics are useful because if current activities are taking significantly longer than the expected time it indicates a prerequisite for this step is missing.

Effect of Reuse: Reuse of existing objects, or portions of newly created objects, has been a cornerstone of the arguments of why you should use an object-oriented approach (analysis, design or language). However, recent articles indicate the measured reuse done in projects is very little.[9] Actually this is not too surprising because conventional elaborative methods focus on design and code reuse of classes, which is very difficult since at this stage in the process the classes have be tailored to a specific implementation. In the Shlaer-Mellor Method, reuse is focused on the analysis models with reuse of entire domains--models of complete subject matter areas.

Reuse of domains occurs when OOA models in a domain are used from one project to the next. It is not uncommon to see 60% to 80% reuse of a domain's OOA models between two different projects. If a project is simply re-implemented on two different software platforms (example, Unix and OS/2), the project would probably see about 100% reuse of the application and service domain OOA models.

Reuse will affect time metrics in that some analysis time will be significantly shorter due to reuse. Time measurements on reused domains need to be kept separate from domains which are newly developed.

Sample Results: Unfortunately few companies collect time metrics on projects. There have been some companies that have collected metrics data on projects using the Shlaer-Mellor Method and have shared their measurements with us. Appendix A contains an example of some these measurements. However, they are like gasoline mileage; your results may vary. They are more useful if viewed in the context of approximate time or relative times for different activities. Project Technology actively encourages groups using the Shlaer-Mellor Method to institute a metrics collection program, and to share their results with other users of the method. It is only with a large pool of data that these measurements will be best understood.

APPENDIX A

Sample Project Results

Project #1: This project was developing an embedded product. The OOA models contained over 200 objects. Their final implementation used C++ and the p-SOS operating system. They collected their statistics using Microsoft Excel by recording the information into a template, and then running a complex Excel macro that collected each analyst's information into a central spreadsheet, subsequently performing statistical analysis on this information to reduce it for reporting.

Observation 1.1: Breakdown of different OOA/RD activities. Performing the different activities of OOA on the project, they measured for relative time percentages:

Partition domains, build Domain chart	2%
Build, review Object Information Models	21%
Build, review State Models	30%
Build, review Action Specification	12%
Total OOA time	**65%**
Develop and test Software Architecture and Translation rules	10%
Manually translate OOA models into code, integrate and test	25%

Shlaer-Mellor expect that 50 to 75 percent of a project's time will be spent performing OOA activities. This project was in the middle of the expected time. They didn't spend much time (2%) on building their Domain chart, which is also expected. Since the Recursive Design translation was performed manually it is not unexpected that 1/4 of the project time was spent in this activity. The translation step could have been automated. Automation of the translation process typically will increase the time spent developing the Translation rules but drastically reduces the translation time to 10 percent or less of the total project time.

Observation 1.2: Time spent performing the modeling. Looking at how much time was spent on developing the models compared to other activities, it was observed as:

Development	77%
Review	12%
Updating	11%

These are expected numbers. A rough rule of thumb is to spend 75 percent of the time developing the models, 15 percent of the time reviewing, and 10 percent updating and correcting the models.

Observation 1.3: Hours per Object. Examining the actual number of hours spent performing OOA and dividing it by the total number of objects they averaged 28.3 hours per object. This was seen as:

Developing OIM	8.6 hours
Developing SM	13.8 hours
Developing AS	5.3 hours

Given that this was for an embedded real-time system it seems reasonable to have the majority of the time spent on state modeling. An information system would probably experience more time spent on the OIM and AS stages.

However, the above measurement is very simplistic since it doesn't differentiate between different patterns of objects. Some objects have no behavior (Passive) and do not require State models and Action Specification and just exist as data while others have behavior (Active) and have State models and Action Specification. In this project roughly 60% of the OOA objects were Active objects.

If we examine the times for these two groups of OOA objects we see:

Activity	Passive Object	Active Object
Developing OIM	8.6 hours	8.6 hours
Developing SM	0 hours	21.0 hours
Developing AS	0 hours	11.5 hours

Observation 1.4: The total time spent on administrative activities (not doing analysis) was 14 % of their time. It was broken down into various activities:

Scheduling and Monitoring the Process	39%
Software Team meetings (analysts and team leaders)	20%
Miscellaneous Administrative	18%
Metrics collection	12%
Software Team leaders meetings	10%

The Scheduling and Monitoring the Process time was large relative to the other times because of the detailed metrics being collected on this project. The collection of metrics allowed the management to monitor the progress and expected completion steps, and to modify the scheduling or personnel assignments if insufficient resources were being applied to a particular part of the project.

Observation 1.5: It took about 1100 hours to develop the Software Architecture and the Recursive Design translation rules. Once this was complete it took about 16.7 hours per object to manually translate, integrate and test the OOA models into code. Again, automation of this step would drastically decrease the time per object to perform the translation.

---

1 Shlaer, Sally and Stephen J. Mellor, *Object Lifecycles: Modeling the World in States*, Prentice-Hall, 1991

2 American Heritage Dictionary 1992

3 Shlaer, Sally and Stephen J. Mellor, *Shlaer-Mellor Method*, White Paper, Project Technology, Inc. 1993

4 Shlaer, Sally and Stephen J. Mellor, "...A Deeper Look," *Journal of Object Oriented Programming*, February 1993

5 Lewinski, David J., *Object-Oriented Development: Metrics For a Defined Process*, Internal paper presented at the Rockwell Software Engineering Symposium, 1993

6 Ibid., *Object Lifecycles*, ref. pg. 154

7 Ibid., *Object-Oriented Development: Metrics For a Defined Process* ref. pg. 9

8 An example of this new generation of tools is the BridgePoint® family by Project Technology, Inc.

9 Firesmith, Donald, Frameworks: "The Golden Path to Object Nirvana," *Journal of Object Oriented Programming*, October 1993