Monitoring the Efficiency of a Reuse Program

L. Benedicenti*, G. Succi#, A. Valerio#, T. Vernazza#

# DIST, Università di Genova, via Opera Pia 13A, 16145, Genova, Italia
* DISA, Università di Trento, via F. Zeni 8, I-38068 Trento, Italia

1. Abstract
Introducing software reuse at a corporate wide level represents one of the most promising means of addressing rising costs plaguing the software industry. A series of mechanisms are needed for shortening development cycles and providing reliable software of high quality which will be more maintainable and flexible for future extensions.

This paper presents an overview of a reuse program that is intended to institutionalise a software reuse process that incorporates reuse-specific activities all along an object oriented software development process, drawing heavily on a reusable software artifact library system which has been designed to support the classification, management and search for artifacts to be employed in reuse efforts.

Due to the increasing competition and rapid technological innovations that mark nowadays the software applications' field, this sector has undergone radical changes, forcing operators to cope with a growing demand of new applications and services, in terms of quality, variety, reliability, and, last but not least, low prices.

In this situation the traditional custom software development methodologies show all its inadequateness and ineffectiveness: to become really competitive in developing new applications, able to survive in a global market, an organisation needs to adopt leading edge technologies inside a defined, well planned and specific software development process.

Moreover, traditional software development methodologies do not allow an efficient maintenance of the products, while any modification or extension of these products often results in a full redevelopment of them.

A new iterative development philosophy based on software reuse and using the emerging object oriented technologies seems to be the right trade-off among the opposed factors characterising a successful telecommunication product or service: high-quality, low-cost, reduced time-to-market, flexibility and maintainability, just to tell some.

In this view, a systematic reuse program to support and to integrate the software development activities, incorporating a reuse library to support the classification and management of the heterogeneous reusable components, is a particularly interesting and promising methodology.

2. Introduction
Software reuse is one of the most promising paradigms in the software engineering field: if it is fully integrated and supported in the software development process, it provides the foundation for dramatically improve the way software is developed and supported over its life-cycle. These benefits include mainly decreased development time and increased product quality and reliability, but also increased productivity, decreased or avoided production and maintenance costs and better use of resources. Reuse fosters modularity and interoperability of applications, even more if integrated with object-oriented paradigms: both reuse and object-oriented paradigms are centered on the concept of asset (or object), i.e. an entity that can be completely and independently modeled and formalized in a given context. Object orientation shifts the development strategy from procedures and functions to objects and interrelations among objects. Object orientation is seen as a more natural and intuitive way to look at and model the real world, which can be considered based on interacting entities [12].

This relatively new technology is rapidly spreading its influence over the whole development process, from design to implementation, and more. As an example, it is the basis of an interesting technique for capturing user requirements in the early stage of software development, proposed by Ivar Jacobson and named Use Cases [2]. In the context of software reuse, object oriented techniques can be seen as a natural prelude to reuse: developing software applications in an object oriented way implicitly allows to identify, model and develop the basic entities which will then constitute the reusable bricks for present and future constructions. Object orientation and reuse freely integrate and support themselves, leading to an improvement of both the software product and process.

A particular aspect regarding reuse concerns testing activities: testing software is one of the
most difficult and time-consuming activity of the whole software life-cycle. Moreover, there are no feasible test techniques that permits to guarantee the software has no defects or bugs. Reusing assets from past projects, assets that have been already tested and proved in-the-field, allows to produce more reliable applications: the test activity that has still to be carried out is focused on the integration phase, while testing of single assets has already been done in past projects when they have been created. Implicitly, each asset must be produced with what could be called 'high quality', and the assets itself must be stored in a repository for future reuse, maintaining and updating it when necessary [3].

Despite the enormous advantages promised, reuse paradigms have encountered many difficulties to be applied in real software development processes. Although reuse promises quality and productivity benefits, observations and researches in the field have shown that reuse needs investments of time and effort, most concentrated at the beginning of the reuse program introduction. Some problems have been reported by different organizations in introducing reuse-centered development processes, not only due to technical issues, but to non-technical aspects, such as collaboration hazzles or legal issues, too [15]. One of the most critical aspect to deal with concerns the transformation of the whole software development process from labor-intensive to capital-intensive: in other words, the high investments requested at the beginning of the introduction of a reuse paradigm can be reclaims only in the long run. Moreover, many organizations that manage to introduce reuse fail in tracking and controlling its evolution, especially compared with organization objectives and mission.

The evaluation of how well organization's goals are being accomplished is based on the selection of an appropriate set of metrics, together with effective measurement practices for collecting the selected metrics. Metrics and measurements practices provide the mechanism for monitoring the status of the software life-cycle, along its development and maintenance. The monitoring activity allows the early detection, and possibly resolution, of inconsistencies and shortcomings inside the software process. This is the only way to support and foster the real integration of reuse paradigm into the existing software development process, encouraging continuous process improvement.

3. A Reuse Program

Introducing a reuse program inside an organization is a multi-faceted and subtle task. Before thinking of the specific reuse program itself, it is necessary to perform an accurate assessment of the current situation of the organization, including the objectives, the mission and the market strategy defined by top management.

The assessment should highlight the state of the software development process: it is clear by now, as well proved by recent studies conducted by [17], that the integration inside the current development process of a reuse program can be effective only if the process itself is well defined and structured, and the software life-cycle is planned and managed inside the organization.

For an organization with a software development process classified at level one of the SEI-CMM [16] it is really difficult to successfully introduce an effective and efficient reuse program.

Another important precondition that must be present before the introduction of a reuse program concerns the identification and description of the specific domain where there will be high reuse potentialities: reuse can significantly increase the efficiency of a development and maintenance processes for applications characterized by commonalities of aspects and targets. This task, usually inserted under the umbrella of Domain Analysis, represents one of the most interesting new field of Software Engineering, but its methodologies are at the moment scarcely diffused and applied.

The general process used to introduce reuse paradigm inside the current development process of an organization, assumed the activities previously outlined have been considered, can be modeled as follows:

1. identify and formalize the primary objectives that the organization wants to achieve with the introduction of reuse;
2. define the aspect and field of the organization which will be interested by the reuse program, choosing the application domains with the greatest opportunities to be reused;
3. identify, inside the chosen domains, the commonalities and collect them;
4. set up a group of people (Reuse Support Organization, RSO) with the task of supporting each reuse activities, developing reuse guidelines and documentation for the organization;
5. define a process to assess the evolution of the reuse program, together with a continuous
process aimed to improve and specialize the reuse program;
6. introduce a strong education and training program to stimulate and support the transition to a reuse-based software life-cycle, fostering the integration of the reuse program in the software development process.

Usually, a reuse program requires the introduction of different support tools, one of which is a Reuse Library (RL). The RL should be able the ability to classify and to store the reusable assets identified during the development for reuse activities, and to search and retrieve the assets that satisfy a given set of properties during the development with reuse phase. The core of the RL is the classification and search mechanism, which should be at the same time efficient and easy to use [13].

4. The Monitoring Program

The main objective of this work is to suggest a general measurement strategy, together with a set of basic metrics, suitable for monitoring the efficiency of a reuse program introduced in the organization. While the monitoring strategy presented could be modified, aiming to support organization specific contexts and objectives, it provides a common starting point for organizations that want to introduce a measurement and monitoring program, enhancing information gathering and management.

4.1 The role of top management and the Monitoring Group

The identification and definition of the objectives of the monitoring program, as well as the definition of organization's main goals, should start from the top management: it is essential that the belief on the importance of the monitoring program originates from the top management and it is spread to all levels of the organization hierarchy, down to the project groups and single employees. Top management should be the leader of the process aimed to define the goals of the measurement and monitoring program, confirming the importance of the program itself. If this lacks, there should be, however, a clear connection between the leader of the group undertaking the monitoring program (which will be called, from here ahead, Monitoring Group) and top management: it should be manifest the importance of these activities to the whole organization.

The definition of the monitoring program should include the identification of roles and responsibilities for each measuring activity, too. The role and responsibility of each employee who is involved in the program should be stated clearly, explaining the relation with the normal activities performed by that employee. The program has to be coordinated by a task leader with the responsibility to organize all the resources involved in the different activities and to respond to the management on the state of the monitoring program and on the results obtained. If it is needed, one or more full-time or part-time employee will support the activity of the task leader and of the Monitoring Group; staff from different projects that are under monitoring, or with specific capabilities, will participate to the program, employing part of their effort gathering measures and data for the program, or generally supporting the Monitoring Group.

4.2 Deriving metrics using the GQM model in the right perspective

The purpose of the suggested monitoring program is to define a set of measurement activities to gather the data needed to verify the efficiency of a reuse program introduced inside an organization.

As previously said, the monitoring program should originate directly from the goals of the organization, inside the more general context of the reuse program introduced. The goals of the monitoring program, as the objectives of reuse, should be periodically revised in order to assure that they remain consistent with the principal goals and the overall evolution of the organization.

Considering these premises, the monitoring program has been derived using the Goal Question Metrics (GQM) model proposed by Victor Basili [14]. The GQM model concerns the application of metrics to the development process using, normally, a three stage process: the first step consists in defining the goals of the measurement activity, then the questions which, if answered, would indicate whether or not the goals have been achieved, are identified. From these questions, just by expressing them in a quantifiable model, come the metrics to be applied. One of the most important benefit coming from following the GQM approach is a structured and organized process for choosing the right metric to apply for achieving the desired goals. Collecting metrics is generally labor intensive: often
it is considered a waste of time in respect to the pure development activity. Moreover, metrics collected are useful only if they are interpretable and usable for feedback actions on the source process where they have been gathered. Metrics should be objective and they should not depend on subjective properties of the personnel who is collecting them (such as experience or skill). The GQM model allows to structure the process of definition, aiming to derive a minimum set of well focused metrics suitable for assessing the goals appointed.

On the other hand, some of the feedbacks coming from industries have led to the understanding that the process of deriving quantifiable questions and choosing the suitable metrics starting from goals is often difficult. This is due to many different factors, both technical and context specific, but it is also a direct consequence of the need to adequate the GQM model to the people who will adopt it.

An effective measurement program should provide different metrics and measurement practices, suitable for the different levels of the organization hierarchy, in order to assess distinct aspects concerning the achievement of the reuse goals established. Therefore, metrics useful for project leaders or developers could be based on the concept of reusable artifacts or reused lines of code, while for the management this data is probably not significant, and, instead, time to market or staff involved in a project could be interesting aspects to evaluate. In other words, the GQM model has to be tailored basing on the specific customer involved in the process, considering the information needed in its work context [11].

One more observation is due: adopting a measurement and monitoring program should be motivated by adequate objectives, and the overhead it introduces in the development process should be carefully evaluated. Measurement activities and metrics collection are time and effort consuming (if they are considered from the point of view of the main objectives of the development process): they absorb resources from the production activity and are often considered useless by programmers. It is fundamental to choose the right set of metrics and measures, capturing whatever is possible without it becoming burdensome, avoiding to collect metrics which are difficult to interpret and use for feedback to the development process.

4.3 The structure of the monitoring program

Having introduced the main problems associated with the definition of a general monitoring program, the strategy proposed in this work for developing and implementing the monitoring program can be synthesized in the following steps:

1. Definition of the organization perspectives for which the efficiency of the reuse program will be evaluated.
2. Definition of the organization goals to be assessed with the monitoring program.
3. Identification of the minimal set of questions whose answers will permit the assessment of the goals.
4. Identification of the core metrics (eventually integrated or modified considering the organization-specific situation) whose evaluation gives the values to answer the previous questions.
5. Definition of the organization measurement plan and data collection activities.
6. Definition and implementation of the organization information system for storing and cataloguing the data derived form the measurement activities.
7. Conduction of an initial assessment in order to define, together with data coming from old projects if present, a metrics ‘baseline’, while validating the measurement procedures and results.
8. Elaboration and interpretation of data collected with metrics for monitoring the efficiency of the reuse program.
9. Continuous improvement of the whole monitoring and measurement process, and modification or specialization of the selected metrics in order to increase their effectiveness in answering to the questions identified in the first steps.

As discussed in the previous paragraph, the GQM model should be applied in the right perspective in order to obtain satisfactory results, but the perspective is strictly correlated to the goals that have to be achieved. Thus, the identification of the goals and the definition of the perspectives should be carried out as a single process. The monitoring program proposed is based on two perspectives, the Top Management perspective and the Monitoring Group perspective, and should achieve the following objectives:

- From the Top Management perspective: evaluating if the reuse program introduced is having success inside the organization, or, in
different words, if it is acting as expected, fulfilling the goals defined previously in its introduction. Top management perspective is necessarily global, considering the whole organization.

- From the Monitoring Group perspective: monitoring the reuse program, gathering the needed data for supplying the information required by the top management: this implies to evaluate the efficiency of the reuse program from an organization-global point of view, integrating and synthesizing information collected from single project interested from reuse (comparing with baseline measures and eventually with projects not interested from reuse). A global monitoring and evaluation of the reuse program is feasible only if information concerning every single project are available: the monitoring group is responsible for mapping the organization-global, integrated perspective goals into more specific goals related to the software development process, from which questions and metrics will be derived. This also means shifting from the top management global perspective to specific perspectives, in particular towards two reuse-related point of view: developing WITH reuse and developing FOR reuse. Goals specific of each perspectives will then originate metrics and measure strategies that will be applied to single projects. Depending on the complexity and work structure inside the organization, the monitoring group assumes different characteristics, roles and responsibilities.

Considering the Monitoring Group perspective, it goes without questions that, deriving the metrics from goals, it is necessary to shift from an organization-global vision to a project-specific vision, while, in the synthesis and evaluation phase, it is requested to join collected measures in order to derive global parameters. This shift of focus from global vision to specific vision and the opposite, that could be configured as a top-down analysis phase followed by a bottom-up synthesis phase, can be associated to the distinction between direct and indirect metrics: the first ones can be measured directly from the object which is subject to measure, while the others can only be calculated from different direct metrics. The top-down analysis phase supports the process of choosing the suitable metrics in the appropriate context to assess the identified goals (following the GQM approach). The bottom-up synthesis phase supports the elaboration of measured metrics aiming to join different obser-

vations to produce a comprehensive solution for a problem, i.e. to assess the goals of the monitoring program. This aspect assumes peculiar characteristics considering software reuse: monitoring and assessment of a reuse program can be done only considering the whole organization, i.e. all software development processes versus time, where the variable time is critical because of the delayed return on investments associated to reuse. In other words, while it is possible to monitor the reuse program nearly runtime at a project level, the assessment of the reuse program and the evaluation of its efficiency is better done (or should only be done) considering the organization in the whole. One more issue arise evaluating a reuse program: considering the development for reuse process, it is not feasible to use traditional measures (with traditional interpretation), such as productivity (for example, barely expressed as size against effort), because of the different target from traditional development process and the extra effort employed. In order to deal with this issue, it has to be introduced a specific separation between development for reuse and development with reuse, not just only because metrics should be different, but because interpretation should take in care different goals and visions.

Considering the above premises, the monitoring program should consider a core set of measures, derived with the GQM approach for the two perspectives considered (Top Management perspective and Monitoring Group perspective).

**Top Management perspective**

Objectives: Achieve demonstrable results from the reuse program, mostly expressed as savings in the software life-cycle costs; verify economic issues of return on investments and cost savings/avoidance due to the reuse program.

Measures:
- **Global vision:**
  - Percentage of software reused (as distinguished from software developed from scratch)
  - Cost against size of software produced
  - Return of investments derived from software reuse
  - Cost saving/avoidance due to reuse (at project level and in the organization as a whole)
  - Software life-cycle cost reduction compared to the estimated ones
Monitoring Group perspective

Objectives: Monitor and verify the reuse program concerning inter-related and specific aspects of software development and maintenance, achieving demonstrable data concerning productivity and quality increase.

Measures:
- **Global vision:**
  - Average effort/resources saving/avoidance due to reuse
  - Average number and type of errors found/corrected during test phase
  - Number of schedule slippages and cost overruns
  - Average development time reduction (against planned)
  - Number of maintenance activities
  - Has the library grown? Is it growing? How much? Has the reuse ratio been augmented? Has the quality been increased?
- **Specific, FOR reuse vision:**
  - Effort spent over size of asset produced in the development for reuse process
  - Effort spent over size of asset modified in the development for reuse process
  - Effort spent over size of asset generalized in the development for reuse process
  - Number of new/modified assets inserted in the RL
  - Number of errors found for each asset produced/modified in the RL
- **Specific, WITH reuse vision:**
  - Effort/time spent in asset identification and search process
  - Percentage of matching assets found during searches
  - Number of assets reused/developed from scratch
  - Effort spent over size of asset specialized in the development for reuse process
  - Size of software reused/size of software delivered
  - Total effort spent, planned effort
  - Number of errors found in the system before delivery

4.4 Storing metrics and measures

An adequate information system should be set up in order to store, analyze and disseminate information inside the whole organization. The information system should be directly coupled with a set of automated tools for the collection of the information and metrics requested, storing all data gathered in a common database. It is not useful to create a stand-alone separate process for collecting and managing the measures, but it should be integrated with the reuse program and the existing development process. In a similar way, the information system should be integrated inside the information system of the organization. Coupling directly the information system with the RL created in the organization by means of the reuse program, many information, such as the number of assets reused inside a project, are gathered automatically from the use of the RL by employee. This data is very interesting and important for the monitoring program, and comes, on the other hand, nearly for free. The RL, first introduced by the reuse program to support the development for and with reuse activity, acting as a repository for storing reusable assets, covers now a new role inside the monitoring program, allowing to automate and to support specific measures activity and metrics collection. The RL becomes an embedded metric tool in the existing environment, allowing to gather many measures without introducing burdensome activities. Moreover, the program is better focused on the process and product, avoiding potential dangerous effects due to subjectivity in collecting metrics of human resources. In order to become the core of the monitoring program, the RL has to be extended and integrated with automated measuring tools and with a suitable information system structure for managing measured data

5. Conclusions

Many organizations that manage to introduce reuse inside their software development process fail in tracking and controlling its evolution, especially as compared with organization objectives and mission. The evaluation of how well an organization’s goals are being accomplished is based on the selection of an appropriate set of metrics, together with effective measurement practices for collecting the selected metrics. Metrics and measurements practices provide the mechanism for monitoring the status of the software life-cycle, along its development and maintenance. The monitoring activity allows the early detection, and possibly resolution, of inconsistencies and shortcomings inside the software process.

This work has presented a monitoring program for evaluating the efficiency of the development process where reuse has been introduced. It has been evidenced the importance of a strong com-
mitment of top management and the issues connected with different perspectives in defining the measurement objectives. The monitoring program has been developed following the GQM model proposed by Basili, and a suitable set of metrics has been identified. Moreover, the infrastructure for the measurement activity has been outlined. Expected results coming from the application of the monitoring program include keeping under control the integration of reuse inside the organization, early detection of possible anomalies or problems, and, more generally, it provides information which are the basis for checking and controlling the whole software development process. The monitoring activity allows to support and foster the real integration of the reuse paradigm into the existing software development process and is the only way to encourage continuous process improvement in the organization.

6. References

Table of Contents

Special Issue on Parallel Object-Oriented Programming

Guest Editors: Boleslaw K. Szymanski, Charles D. Norton

M. Austern, R. Towle, A. Stepanov:
Range Partition Adaptors: A Mechanism for Parallelizing STL

S. Baden:
Software Infrastructure for Non-Uniform Scientific Computations on Parallel Processors

D. Gannon, S. Diwan, E. Johnson:
HPC++ and the Europa Call Reification Model

Y. Ishikawa:
MPC++ Approach to Parallel Computing Environment

A. Chien:
ICC++ - A C++ Dialect for High-Performance Parallel Computing

C. Kesselman:
High Performance Parallel and Distributed Computation in Compositional CC++

C. Norton, V. Decyk, B. Szymanski:
On Parallel Object-Oriented Programming in Fortran 90