OptimalSQM: Optimal Software Quality Management Framework Architecture

LJUBOMIR LAZIĆ, EJUB KAJAN
Technical Faculty, State University of Novi Pazar
Vuka Karadžića bb, 36 300 Novi Pazar,
SERBIA
llazic@np.ac.rs, ekajan@np.ac.rs, http://www.np.ac.rs

NIKOS E MASTORAKIS
Technical University of Sofia,
English Language Faculty of Engineering
Industrial Engineering, Sofia, BULGARIA
http://www.wseas.org/mastorakis

Abstract: - A Software companies are often challenged with providing QA and testing of their software in an effective and efficient manner. What a QA staff and project management must do to make software quality more certain? They must have adequate software project management and test management infrastructure such is Optimal Software Quality Management (OptimalSQM) Framework. OptimalSQM Framework consists of a Business Intelligent Simulation Architecture with integrated Software expert tools (Profit eXpert, Planner eXpert, Risk Management eXpert, Quality eXpert, Maintenance eXpert, People Performance eXpert and Process Dynamics Control eXpert) in order to find optimal development activity combination at the beginning for every SDLC model. OptimalSQM solution is an faster, better and cheaper solution which enable software designers to achieve a higher quality for their design, a better insight into quality predictions for their design choices, test plans improvement using Simulated Defect Removal Cost Savings model as we described in this paper. In this paper we described OptimalSQM Framework Architecture, proposed a set of components which enables to minimize the cost of switching between test plan alternatives, when the current choice cannot fulfill the quality constraints.

Key-Words: - Software Quality, Testing optimization, Cost of testing Simulation, Test management

1 Introduction

Software reviews, inspections and testing, as SQA (Software Quality Assurance) activities, are key assumptions for successful software development and implementation. Software industry spends more than a half of its budget for test activities during software development process and for its maintenance after installment. Development of high quality software is very complicated and unreliable task, but the management of software development and testing process (SDP-STP) is much harder without appropriate software environment consisting of integrated techniques, procedures and tools for: a) precise planning (resources, costs, duration, training, etc.), b) software project risk identification, estimation and control, c) software quality metrics and measurement establishment, d) process testing quantitative management, i.e. management of activities needed for software quality assurance in order to increase efficiency of software bugs detection and removal. [1-4].

Our research [5,6] concluded that existing approaches for assessing and improving the degree of early and cost-effective software fault detection are not satisfactory since they can cause counter-productive behavior. An approach that more adequately considers the cost-efficiency aspects of software fault detection is required.

This claim is evident if we take a look into known document – „Chaos Report“, by USA Standish Group, 2001, based on analysis of 8,000 large projects. This report found that 25% of them

1 This work was supported in part by the Ministry of Science and Technological Development of the Republic of Serbia under Grant No. TR-35026.
have never finished due to significant costs and duration exceed, as well as bad quality or the combination of that. In addition the analyses of finished projects shown that average costs are 90% exceed estimated costs, while their duration were 120% longer. Another research, made by USA NIST (National Institute for Standardization and Technology) in 2002 [8], based on very wide sample, pointed out that development of appropriate software environment for SDP-STEP, as mentioned above, may result with significant costs reduction (in sampled economic brunches between $600 million and $1,500 millions).

Avoidable rework consumes a large part of development projects, i.e. 20-80 percent depending on the maturity of the organization and the complexity of the products [9]. High amounts of avoidable rework commonly occur when having many faults left to correct in late stages of a project. In fact, research studies indicate that the cost of rework could be decreased by up to 50 percent by finding more faults earlier [2, 6, 9]. It might appear easy to reduce the amount of rework just by putting more focus on early verification activities, e.g. reviews and inspections [2]. Therefore, the interest from industry to improve this area is large.

What a QA staff and project management must do to make software quality more certain? They must have adequate software project management and test management infrastructure such is Optimal Software Quality Management (OptimalSQM) Framework which is depicted on Fig. 1.

**Fig. 1 Optimal Software Quality Management (OptimalSQM) Framework**

OptimalSQM Framework consists of a Business Intelligent Simulation Architecture (BISA) with integrated Software expert tools (Profit eXpert, Planner eXpert, Risk Management eXpert, Quality eXpert, Maintenance eXpert, People Performance eXpert and Process Dynamics Control eXpert) in order to find optimal development activity combination at the beginning for every SDLC model. OptimalSQM solution is an faster, better and cheaper solution which enable software designers to achieve a higher quality for their design, a better insight into quality predictions for their design choices, test plans improvement using Simulated Defect Removal Cost Savings model as we described in this paper. In this paper we described OptimalSQM Framework Architecture, proposed a set of components which enables to minimize the cost of switching between test plan alternatives, when the current choice cannot fulfill the quality constraints. We offered a simulation method with which it is possible to assist the test manager in evaluating test plan alternatives and adjusting test process improvement decisions in a systematic manner.

### 2 Goals and benefits of the OptimalSQM Framework research areas

Developing software is hard enough but managing software development projects is even harder to do ‘by eye’. Tools are needed to show your development process as it really is: in project execution, in plans and bids and in comparison to other projects in the market. Organizations developing significant amounts of software are often challenged with providing quality assurance and testing of their software in an effective and efficient manner [4,6,9,10].

Managers should be focused at all times on the goal of their project and the risks that threaten them. Projects need to manage goals achievement through valuable information provided, for example, by our OptimalSQM solution that enables management to have influence over the way projects are tested and how the gathering of intelligence will be performed, not from activities to ensure work is done according to bad plan. Intelligence is gathered throughout the project lifecycle, from early reviews to final acceptance testing with integrated software expert tools similar to ours integrated in the OptimalSQM Framework. The intelligence gathered, enables project and stakeholder management to judge how and whether progress is being made. It dramatically enhances the value of testing to their projects and
enables them to manage their projects with more confidence, better visibility of progress and enables them to make significantly better-informed decisions [5,6,10,11].

What we do know is that we want to incorporate in our product planning such qualities as expandability, flexibility, integrity, interoperability, maintainability, portability, reusability, resilience, and usability. These qualities cannot, in general, be "inspected in" or "tested in." They must be "designed in," or more specifically, brought in during the early phases of development i.e. requirements capture, analysis, architecting, and design. During this "design in" process they must be weighed against users' needs in an iterative feedback loop.

2.1 The OptimalSQM Framework

Architecture solution is a Software Testing Center of Excellence

The OptimalSQM solution aims to assist with advanced software testing technology adoption in organizations, especially focusing on small- to medium-sized enterprises (SMEs) i.e. OptimalSQM solution should become a Software Testing Center of Excellence for SMEs as we published in our works [5,6,11]. One of the aims of our project is to set up a scalable, adaptable, reconfigurable facility for an independent validation and conformance process for existing commercial software testing tools to address industry's current concerns as articulated by SMEs i.e. not to develop and implement software but to assist in intelligent software project management.

What are the benefits of a SMEs Software Testing Center of Excellence?

A Software Testing Center of Excellence provides:

• **Scalability** – ability to scale software quality and testing services to changing demand and still maintain adequate quality of service.

• **Objectivity** – holds development accountable for the software they build by providing independent assurance of software quality.

• **Consistency** – standardizing on testing best practice process and tools guarantees that software testing activities will be performed in a consistent and repeatable manner.

• **Constant Improvement** – on-going training, measurement/metrics, and process improvement assure continued strides toward a best of breed software testing capability.

• **Better, cheaper, faster products** – objective, full lifecycle software assurance following standards and best practices reduces software development rework as critical defects are found earlier in the process and corrected.

2.2 The OptimalSQM solution includes a Planner eXpert

Our OptimalSQM can provide (applying best estimation methods [12]) an adequate amount of schedule time for software development, sufficient staff hours, and the productivity to enhance that time and effort. It can provide a software development process in which the key actions that assure quality are given early emphasis. You can see that providing "adequate" amounts of these ingredients is just another way of saying that quality rests upon the ability to employ the metrics for time, effort, size, productivity, and quality judiciously. Starting at the requirements phase (but usually needing to be repeated at each major subsequent review) managers must plan projects by predicting necessary cost and effort and assigning resources appropriately [5,12].

Doing this accurately has become one of the 'holy grail' searches of software engineering. The desire by managers for improved methods of resource estimation provided one of the original motivations for deriving and using software measures. As a result, numerous measurement-based models for software cost and effort estimation have been proposed and used. Examples include Boehm’s COCOMO model, Putnam’s SLIM model and Albrecht’s function points (FP) model [3,4,7].

These models share a common approach: effort is expressed as a (pre-defined) function of one or more variables (such as size of the product, capability of the developers and level of reuse). Size is usually defined as (predicted) lines of code or number of function points (which may be derived from the product specification). There is no definitive evidence that using models such as these does lead to improved predictions. However, the predictive accuracy of all of the models is known to improve if a data-base of past projects (each containing the same variables used in the models, such as effort and size) is available [12]. The availability of such a database can lead to reasonably accurate predictions just using standard statistical regression techniques. This suggests that the models may be redundant in such situations, but the models may still be extremely useful in the absence of such data. Moreover, the models have had an important historical impact on software metrics, since they have spawned a range of measures and measurement techniques which have impacted on QA activities.
far removed from resource estimation. Resource estimation (and indeed post-project process improvement planning) can only be done effectively if something is known about the productivity of software staff [13]. Thus, the pressing needs of management have also resulted in numerous attempts to define measures and models for assessing staff productivity during different software processes and in different environments.

2.3 The OptimalSQM solution includes a Quality eXpert

The OptimalSQM can provide testing managers tool to improve SDP-STP via SQA alternative plans using advanced simulation models of defect detection and removal process effectiveness and efficacy, necessary resources for optimized SDP-STP as we published in our works [6,10,11]. Good metrics [5] and instruments made it possible to plan a journey accurately in advance and to execute that plan with the flexibility to adapt quickly and effectively to unexpected circumstances. These products give you an insight into the performance levels in the market and an objective view of your own performance. When combined, these products allow you to effectively manage the risks involved in software development projects. In addition, The use of measurement instruments (integrated in the OptimalSQM solution) for company projects has both a major and a positive impact on the companies using them.

Measurement is the knowledge and as the knowledge about the work increases there is the beneficial effect that the staff involvement increases also. The goal of all software development activity is where the plan works through to completion, the project delivers to the plan and the company is at all times focused on delivering the project to meet the ‘all important’ client expectations.

The basic demanding elements we deployed in our OptimalSQM framework solution, to fulfil goals that are the Carnegie Mellon Software Engineering Institute (SEI) recommended on core software measures, namely software size, time, effort and defects to reach 4th And 5th Capability Maturity Model (CMMI) level and compliant assessment method for the Test Maturity Model Integrated (TMMI).

2.4 The OptimalSQM solution includes a Profit eXpert

The OptimalSQM exploited well known Balanced Productivity Metrics (BPM) which incorporates the use of both quantitative and qualitative data for measuring performance and productivity improvement [5]. The primary goals of a SEI-CMM based process improvement program are to improve productivity, improve quality, and reduce risk via consistent processes. BPM focuses on the SEI CMM core measures (size, time, effort, and defects), as well as other data collected to measure process improvement. A multi-dimensional approach is needed to measure productivity improvement in a way that also provides an understanding of the environmental factors (and other significant factors) that influence project productivity [11]. The BPM approach is used to measure productivity with regard to software development and maintenance. BPM is based on the principle that the management of productivity improvement should focus on achieving a balance of time (schedule), cost (effort), and quality (defect rate) named TCQ improvement [6]. It is consistent with a balanced scorecard philosophy, since these fundamental metrics components cannot be observed independently with regard to providing a valid productivity assessment. The OptimalSQM solution focus is on building economic models to make the business case that software metrics can really reduce costs and have an impact on the bottom line as we described in our works [5, 10].

The OptimalSQM solution will do this by being precise about how metrics (ROI, BCR, CAPEX, OPEX etc.) can help inform very specific decisions, and then evaluated the economic benefit of making the optimal decision [10,11]. Many software companies were reducing their expenditures on quality assurance and quality control — they had to reduce costs to survive and quality was one of the first victims. But also, security vulnerabilities became a major issue of concern to governments and enterprises. Security vulnerabilities are a manifestation of low quality. There was a need for more QA and QC rather than less.

2.6 The OptimalSQM framework deployed Six Sigma strategy

Our OptimalSQM framework solution focuses on a significant part of software testing process improvement effort which pertains to defect prevention, software testing technology change management and software testing process change management, besides of productivity improvement based on our current published research results [16] that three years of proposed software environment deployment with integrated software tools in OptimalSQM will result with ROI of 100:1, comparing with existing SPD-SPT infrastructure of software companies that have achieved CMM and
TMM maturity of the 1 and 2 level. The OptimalSQM framework implement Six Sigma strategy in Intelligent Process Improvement (iPI) solution which must combine TCQ and Six Sigma approach. It should provide an automated-test-tool evaluation and selection technology developed to help SMEs improve their management of software testing technology change on intelligent way. It makes sense that before you can improve a process you understand fully where you are and what needs to be improved. Although the OptimalSQM has software-process-improvement oriented application, it can also be used to help forensic software engineers more easily identify candidate equipment for software-intensive incident and accident investigations. The problem with traditional automated-test-tool (or more generally, computer-aided software engineering (CASE) tools) evaluation and selection techniques is that they provide limited visibility/measurement into the selection (acquisition and/or equipping) of automated-test-tools (or CASE tools). In forensic investigations of software-intensive accidents and incidents, it is important that forensic software engineers correctly identify, measure, and collect the data needed to draw valid conclusions regarding technology adoption on economic way.

2.7 The OptimalSQM solution includes a Testing Dynamics Control Model

The OptimalSQM solution to be practically useful for determining which activities need to be addressed to improve the degree of early and cost-effective software fault detection with assured confidence, than definitely, optimality and stability criteria of very complex STP Dynamics Control Model of SDP-STP (more than 100 variables are identified) described in Fig. 2 must be established [11]. A basic rule from cybernetics - that a long time lag between the output signal from the controlled system and feedback to the controller causes instability in the system - applies to SDP-STP processes as well. Long design iteration loops with late feedback drive cost and schedule overruns in SDP-STP.

2.8 The OptimalSQM solution deployed Risk Based Testing Optimization Model

The OptimalSQM solution focus is on the software reliability, because customers rank reliability first on their list of most critical quality attributes, according to a recent survey of the largest customers of complex real-time telecommunications systems and data published in the software metrics literature [2-4]. With the cost of some systems exceeding tens or even hundreds of millions of dollars and with a development duration of more than 12 to 18 months, early reliability estimation can significantly contribute to the success (or early rational cancellation) of the project. With recent strong emphasis on speed of development, the decisions made on the basis of early reliability estimation can have the greatest impact on schedules and cost of software projects. Software reliability may also be significantly improved in an early stage by a focused review and inspection process, early defect removal, and thorough test effort.

To increase estimation accuracy and our ability to better control the discovery-of-faults process, our partially done research of software reliability issue [14,15] concluded the following: The software reliability estimation in the OptimalSQM framework solution must be performed in early phases of the life cycle by using phase-based models that emphasize the availability of size and corresponding effort for the project during early phases. Software reliability estimation provides a solid foundation to perform meaningful tradeoffs studies at project start. It also provides a projection of the software failure rate before systems tests begin or at any point throughout. After estimation, the next logical step is creating a software reliability growth model, which covers the period where reliability improves as the result of thorough peer reviews, testing, and fault correction. Reliability metrics [5,14,15] help to predict such critical factors as the initial failure rate, final failure rate, fault density, fault profile, etc.

Fig. 2 The feedback control model for SDP-STP

The final outcomes of software reliability estimation include the following:
Recent Researches in Software Engineering, Parallel and Distributed Systems

1. Benchmarking and Process Improvement
2. Estimating and Risk Assessment
3. Progress Control and Reporting

Process improvement enables the same amount of software to be built in less time with less effort and fewer defects. Informed estimating uses process productivity benchmarks to evaluate constraints, assess risks and to arrive at viable estimates. Estimates of the defects at delivery use the history from benchmarked projects and allow alternative staffing strategies to be evaluated. Throwing people in to meet tight time to market schedules has a disastrous impact on quality. Progress control tracks defects found during development in order to avoid premature delivery and to ensure the reliability goals are achieved.

Each process contributes separately to improving the quality of the final software product. Dramatic quality improvements are achieved by dealing with all three. SQA is defined as “a group of related activities employed throughout the software lifecycle to positively influence and quantify the quality of the delivered software.” The first area, benchmarking and process improvement is a self-evident need for every software development organization concerned to demonstrate value for money and to reduce the software lifecycle time and effort as well as the number of defects.

Benchmark results enable estimates to be made for new developments based on known process productivity and also provide the means to check these estimates against the benchmark history. Purchasing organization need to benchmark their software suppliers (including outsourcing suppliers) to establish contract baselines and build solid evidence of supplier on-going process improvement.

Software estimating and risk assessment is fundamental given the well-documented evidence of continuing software disasters characterized by cancelled projects as well as horrific cost and schedule overruns coupled with poor delivered quality. In-progress control and reporting of development progress gives continuous visibility to both developers and purchasers. Time to market pressure in many high technology domains’ means that developers and purchasers alike require at all times to be confident the software delivery date, the expected cost and reliability will be achieved as planned. Benchmarking and process improvement quantification confirms that real commercial benefits are being achieved through improved development productivity. Project managers in development and purchasing ensure realistic estimates are made consistent with known

• An estimation of the number of faults expected during each phase of the life cycle as we published in our works [6,11].
• Constant failure rate estimation at system release [11].
• Relative measures for practical use and management such as duration of system test and size of the test team [12].

If software reliability estimation is performed early in the software life cycle, it is possible to determine what improvement, if any, can be made to the software methods, techniques, or organizational structure. For early software reliability estimations (SRE), an estimation team must use a number of software reliability models. The team must compare the results of these models with each other and with the reliability data provided by cost/effort estimation model in the OptimalSQM solution, which will be selected after study of more than a dozen different estimation models (the COCOMO tool came out a very close second). Our Risk Management eXpert tool in the OptimalSQM solution uses large database (from our database and data published in literature consisting of more than 8,000 actual projects [4]), that along with size, cost, effort, and other attributes have data about the total number of inherent faults, their distribution among major phases of life cycle, the severity of faults, and potential defect removal efficiency.

As with any other existing cost estimation tool, SRE in the OptimalSQM solution have possibility to be fine-tuned and thoroughly calibrated for the particular project to be consistent and reliable. At the same time in our BISA (Business Intelligent Simulation Architecture) environment, the SRE questionnaires on the project/process development activities should be used as a shell for the company-wide software process assessments, together with some parts of the SE Capability Maturity Model® (CMM®). These assessments should be used for internal benchmarking of current development practices and helped to easily identify similar projects, which was essential for a meaningful comparison. In the OptimalSQM solution Risk management of proposed SDP-STP is fully integrated to perform meaningful tradeoffs studies at project start [12,14,15]. Here some facts will be emphasized. The role of SQA, as described here, is to confirm that core measures are used effectively by management processes involving technical and purchasing directors, project and purchasing managers and process improvement groups. They cover:

1. Benchmarking and Process Improvement
2. Estimating and Risk Assessment
3. Progress Control and Reporting
constraints and the benchmarked process productivity. Equally important is to quantify the estimate uncertainty and risk [14,15]. A regular procedure quantifies improvement benefits [6]. At intervals, say every 6 months, a report must be made showing benefits from recent projects through initiatives such as CMMI [11]. This includes calculating the Return on Investment (ROI) based on productivity improvement and investments made to improve [10].

In a development group SQA is performed on the documented estimating procedure, as we planned in the OptimalSQM solution, to check input data is used that quantifies:

- The software product size and uncertainty
- The development process productivity
- The development constraints: time, effort, staff, and reliability
- The risk levels for each constraint

Software acquisition in the OptimalSQM solution should require the supplier to provide this data using a formal questionnaire. Software size is quantified using the estimated size range. Each software module is estimated in terms of the smallest, most likely and largest size. This size range uses the most practical sizing units such as logical input statements, function points or objects [12]. The time, effort, resources, costs and reliability constraints for the development are risk assessed taking into account the quantified uncertainties such as the size range. Each constraint must be associated with a risk level and estimates should be evaluated against these risk levels in the OptimalSQM solution. Frequently it is found that specific constraints cannot be met since the risk is too high. The estimating procedure should evaluate alternatives, each of which is logged and documented. This may mean allowing additional time, adding staff and/or reducing features (size) [11,16]. The alternative “What If” estimates document how the final baseline plan will be determined, risk assessed and agreed as we planned to fulfil in our BISA (Business Intelligent Simulation Architecture) environment via SQA alternative plans assessment using advanced simulation models of defect detection and removal process effectiveness and efficacy deterring necessary resources for optimized SDP-TP.

3 The Concept of the OptimalSQM Framework Architecture

In order to realize software environment for optimized quality software development (faster, better and cheaper than other solutions), the research in this TR-35026 project will focus on the software environment requirements via further evolution of BISA, partially defined in [6,11] (see figures below that represent main software components of possible BISA solution). Such BISA will allow software designers to achieve higher quality of design and better view for its estimation quality. In addition testing managers will be able to improve SDP-TP via SQA alternative plans using advanced simulation models of process detection and defects of necessary resources removal [6,11]. BISA kernel should be unified, coherent, balanced and improved version of OptimalSQM, providing the full service spectrum that satisfy the highest maturity level (4 and 5) of SPD-TP according to SEI CMM and TMM methodologies. Here is short description of BISA concept level. BISA consists of 5 components: MGR, OPS, POST, SIM and BOX, available as a comprehensive suite of test management solutions or modularly, as building blocks as shown on Fig. 3-7.

![Fig. 3 The BISA - main software components (MGR, OPS, POST, SIM and BOX) and MGR’s sub-components](image-url)

The MGR component is at the heart of BISA, providing integrated and coherent management of the multidisciplinary facets of testing operations, enabling advanced generation of testing rules. MGR incorporates SaaS’ (Software as a Service) paradigm
Rules – that will be the industry’s first software testing scenario language with easily customizable predefined rule templates – to address critical test management vectors. Also, important function of MGR component is to provide all questionnaires on the project/process development activities and process productivity benchmarks to evaluate constraints assess risks and to arrive at viable estimates, of specific company and project. The SIM component will be a new industry capability that enables the simulation of software testing rules on your actual test results repository prior to implementation in production. SIM helps project management quantify benefits, schedule monitors and execute what-if scenarios for maximum efficiency and best ROI. The BOX component will be the industry's best practices, universal software test techniques repository for all testers, provers, handlers, and purchased software test tools. The BOX will be totally process and device independent, supporting all levels of test parallelism. As part of OptimalSQM solution, it executes rules that have been created and simulated and implements them for all SDLC test activities and final test.

The OPS component integrate a set of centralized test operations solutions that enable a Software Testing Center of Excellence for SMEs. It supports the need for productive and efficient test management while pinpointing inefficiencies to enable prompt responses. OPS delivers complete planning, control, execution, and monitoring of the entire software testing activities sort and final test operations and reports.

The POST component re-evaluates all testing results to better control the quality and health of all facets of your testing operations. POST performs cross-quality evaluations of all testing fleet, and all test efficacy evaluations and defect detection and removal yield, offering extreme data-integrity. Furthermore, the POST augments software reliability through SRE (Reliability metrics to predict and estimate critical factors as the initial failure rate, final failure rate, fault density, fault profile, etc.) and other advanced detection techniques selection exploiting Design on Experiments knowledge. Based on these data POST provides after software product release complete technical support i.e. Maintenance program for Corrective, Adaptive, Perective and Preventive maintenance activities on optimized way. Basically, BISA is a SOA based on SaaS paradigm consisting of several expert tools as follows: a) Profit eXpert for analyzing different scenarios for project realization based on estimation models, software volume and complexity prediction, development and testing duration, in order to choose the best scenario for the project realization; c) Risk Management eXpert together with Profit eXpert provides software development and testing managers ability to identify and estimate effects and to reduce risks on an acceptable level.

Fig. 4 The BISA main software components: OPS and POST
The role of Quality eXpert is to integrate specialized software tools on demand (Quality Metrics eXpert, Test Effort Estimation eXpert, Reliability eXpert, Product release eXpert), in order to provide software managers with the ability to build appropriate software quality metrics, automate software process, etc. based on the same parameters that are used by Planner eXpert.

Maintenance eXpert should provide service to software managers with the ability to plan and estimate costs of corrective, preventive, perfective and adaptive software maintenance. As emphasized previously, the development of quality software is a very complicated and unreliable task, but the management of software development and testing (with more than 100 variables) is much harder without appropriate software environment. From that point of problem, Dynamics Control eXpert should allow identification observable and controllable variable for a given software project and to establish reliable and optimization criteria for every phase of SPD-SPT, as well for the whole software process. Finally, People Performance eXpert takes care of human resources on a software project in order to track and improve their professional skills.

The BISA realization with integrated expert tools (Profit eXpert, Planner eXpert, Risk Management eXpert, Quality eXpert, Maintenance eXpert, People Performance eXpert and Process Dynamics Control eXpert) will offer to SMEs the following: 1) Web portal-repository of the best models and techniques from practice, all integrated into optimized and quantity driven software testing and maintenance process; 2) an environment for scenario of qualitative software development simulation that allows cost and risk minimization by alternative testing plans choice, which satisfy constraints related to free resources, optimization criteria and performances of a company; 3) economical model of software quality for SQA activities evaluation of the investment benefit, measures that should be taken in order to improve software process development and testing (SPD-SPT) based on economical parameters (ROI, BCR, CAPEX, OPEX, etc.).
various abstraction levels of the software under test (SUT) to manage stable (predictable and controllable) software testing process at lowest risk, at an affordable price and time.

Fig. 7 The BISA main software component POST and its sub-components

When we finish the OptimalSQM solution it will guide small- to medium-sized enterprises (SMEs) to adopt rigid standards for verification and validation (TMMI level 4 and 5) of deliverables throughout the software life cycle. It requires a rigorous test methodology with test specifications that are traceable to system and business requirements. Our OptimalSQM solution should become a Software Testing Center of Excellence for SMEs to:

1. Integrate testing into the entire development process (Inspections, walkthrough, unit test, integration test, system test, acceptance test, performance test, security test, load and stress test etc.)
2. Implement test planning early in the life cycle via Simulation based assessment of test scenarios
3. Automate testing, where practical to increase testing efficiency
4. Measure and manage testing process to maximize risk reduction
5. Exploit Design of Experiments techniques (optimized design plans, Orthogonal Arrays etc.)
6. Apply Modelling and Simulation combined with Prototyping
7. Continually improve testing process by proactive, preventive (failure mode analysis) Six Sigma DMAIC model

Also, the OptimalSQM solution will provide the full Project support services spectrum that satisfies the highest maturity level (4 and 5) of SPD-STP according to SEI CMM and TMM methodologies such:
1. Earned Value Management
2. Quantitative Project Management (size, cost, effort, resource, schedule planning)
3. Risk Management (Project Risk Assessments which task is to identify the critical risks of a project and provide detailed guidance on how the risks can be reduced. Testing resources are focused where they will do the most good)
4. Software Reliability prediction and estimation
5. Custom P³ Measurement process deployment (Product, Project and People) i.e.
   • Key Product quality measures including sizing, coverage, defect management,
   • Project progress measures, and
   • People - project team performance measures, are used to quantify the process effectiveness, efficacy and identify risk

It is estimated, based on our current published research results [5,6, 10-16], that three years of proposed the OptimalSQM software environment deployment with integrated software tools will result with ROI of 100:1, comparing with existing SDP-STP infrastructure of software companies that have achieved CMM and TMM maturity of the 1 and 2 level.

4 Conclusion

Testing is one of the most resource-intensive activities in software development and consumes between 30 and 50% of total development costs according to many studies. We described the novel OptimalSQM framework architecture that can provide testing managers tool to improve SDP-STP via SQA alternative plans using advanced simulation models of defect detection and removal process effectiveness and efficacy, necessary resources for optimized SDP-STP. The BISA kernel of the novel OptimalSQM framework architecture realization with integrated expert tools (Profit eXpert, Planner eXpert, Risk Management eXpert, Quality eXpert, Maintenance eXpert, People Performance eXpert and Process Dynamics Control eXpert) will offer to SMEs the following: 1) Web portal-repository of the best models and techniques from practice, all integrated into optimized and quantity driven software testing and maintenance process; 2) an environment for scenario of qualitative software development simulation that allows cost and risk minimization by alternative
testing plans choice, which satisfy constraints related to free resources, optimization criteria and performances of a company; 3) economical model of software quality for SQA activities evaluation of the investment benefit, measures that should be taken in order to improve software process development and testing (SPD-SPT) based on economical parameters (ROI, BCR, CAPEX, OPEX, etc.).

References: