Abstract – In order to enable quality impact prediction for newly added functionality to the existing telecommunications system, we introduce new modeling elements to the common development process. In the paper we describe a black-box and grey-box instrumentation and modeling method, used for performance prediction of a distributed system. To verify basic concept of the method, a prototype system was made. The prototype consists of the existing proprietary telephone exchange system and new, open-source components that make an extension of the system. The paper presents conceptual architecture of the prototype and analyzes basic performance impact for introduction of open-source software into existing system. Instrumentation and modeling parameters are discussed, as well as measurements environment. Initial comparison of predicted and experimentally measured values validates current work on the method.

Keywords – black-box modelling, performance prediction

I. INTRODUCTION

Current software trends demand rapid product development and support. Faster development yields error-prone product what is not acceptable for telecommunications system which is intensively used product consisting of both hardware and software. During the product life cycle and the phase of introducing new functionality, it is necessary to maintain high level of product quality characteristics, like performance and availability, on top of essential functionality.

In order to keep and predict the product quality characteristics, while assuring faster development and functional updates, it is necessary to modify existing development process. Thus, we decided to test a modification of our development process with introduction of new modeling elements into the process. These modeling elements provide quality impact prediction for developed system as a whole, and for any of its parts. Such prediction is important in the early stage of system specification, before the start of intensive development activities.

Essential method and tools that can consistently support performance prediction in the early stage of system specification are our main motivation for participation in the Q-ImPrESS project (Quality Impact Prediction for Evolving Software Systems) [1]. The goal of Q-ImPrESS project is creation of a new software engineering method and tools for enabling creation and evolution of software with end-to-end quality. Currently available methods and tools within the project enable us to perform a basic test of the usefulness of modeling and quality impact prediction in common development process.

Our test bed for quality impact prediction consists of a prototype where legacy system's functionality is extended through non-legacy subsystem. In our case, the legacy system is an telephone exchange that is based on proprietary technologies and mostly on proprietary signaling protocols. The non-legacy subsystem is a widely accepted and known DIAMETER protocol [2] for achieving authentication, authorization and accounting (AAA) [3]. Additionally, the non-legacy subsystem is based on open-source software. Thus, with the prototype we test a scenario of rapid update of an existing product with an open-source based components that add additional functionality to the product.

In order to maintain the legacy system's characteristics like performance and high availability, we model both the legacy system and the new subsystem. Performance analyses derived from the models give enough insight into resulting system's behavior before the actual non-legacy subsystem is implemented and interconnected with the legacy system.

Due to proprietary nature of the legacy system, it is not efficient to model it by reverse engineering or to model it in fine grain. Thus, we use our expertise of the legacy system to model it as an interconnection of a few black-boxes. Under the term black-box, we consider a component having only interface information. Performance parameters that describe a black-box are obtained by instrumenting parts of the legacy system that comprise corresponding black-boxes.

Since the non-legacy subsystem is open-source based, it can be treated like a collection of white-boxes, i.e. components with available source code. Such white-boxes can be reverse engineered into a model. However, models derived by reverse engineering are too fine grained and not feasible for quick design experimentations and analyses. Thus, we use a different approach. We apply light-weight daemons that monitor relevant performance parameters of executed open-source solutions. Information gathered in such manner gives insight in open-source solution's behavior. These information serve as a basis for creating a model of the non-legacy
subsystem. Hence, we do not treat the subsystem as a collection of white-boxes or black-boxes, but a collection of gray-boxes [4]. Gray-box can be instrumented in a simpler manner than a black-box, usually by calling specific APIs and monitoring counters offered by the operating system.

The rest of the paper is organized as follows. The second section presents the related work regarding black-box and grey-box instrumentation and modeling. The third section describes the architecture of the prototype and applied instrumentation and modeling. The fourth section compares results from real-world prototype experiment and results obtained by performance prediction from the model. The fifth section states the conclusion of the paper.

II. RELATED WORK

There are few research projects which relate to our work. In fact, we integrated some experiences from these projects into our project without intrusion to specifics of our work.

In accordance with the study of instrumenting black-boxes and gray-boxes [4], we use typical unobtrusive black-box instrumentation for performing measurements on the legacy system. In this case we gather information about the legacy system through external observation, without interfering into the legacy system's code or even operating system. The information is gathered through operating system's kernel logs, which are commonly used for testing and debugging purposes. Furthermore, the non-legacy subsystem is instrumented as a typical gray-box. This means we use all the APIs offered by the operating system for collecting performance information.

We employ a targeted profiling principle in regards to black-box and gray-box instrumentation as described in the study of profiling and instrumentation techniques [5]. Due to our previous work on the legacy system, we acquired deep knowledge of it, and thus, know how to test it efficiently with a subset of all possible test parameters. Full profiling, with all possible test parameters, isn't feasible because of the legacy system's complexity and time constraints. On the other hand, mentioned study shows equal coverage of full profiling and properly conducted targeted profiling.

Black-box and gray-box modeling used in this paper, can be observed as data-driven modeling of an input/output phenomenon [6]. In contrast to modeling described in the study, we possess significant a priori knowledge of the observed legacy system, as well as solutions constituting the non-legacy subsystem. This enables us with simplified, yet more accurate system instrumentation and modeling. We further increase the quality of modeling by complementing the input/output modeling with traffic emulation [7], where traffic and load imposed on black-boxes and gray-boxes corresponds to real-world traffic and load conditions.

III. THE PROTOTYPE

A. Conceptual architecture

Conceptual architecture of the prototype is presented in Figure 1. The prototype provides DIAMETER-based AAA functionality to existing telephone exchange system. For example, this functionality is requested when an incoming SIP or H.323 call needs to be authorized or when an Internet user needs to be authenticated.

The proprietary legacy system consists of an AXE telephone exchange node [8]. The non-legacy subsystem is comprised of two clusters which assure high levels of performance through load balancing, and availability through redundancy. Clusters are built of standard PCs. Each cluster represents a side in the DIAMETER protocol. Client cluster and server cluster communicate via redundancy. Clusters are built of standard PCs. Each cluster represents a side in the DIAMETER protocol. Client cluster and server cluster communicate via DIAMETER protocol.

Client cluster is connected to AXE node, from which receives AAA requests, and to whom it sends corresponding responses. Client cluster and AXE node communicate via proprietary TCP-based AAA protocol. The proprietary protocol is used because of its existing support in AXE node, and its simple implementation in the non-legacy system.

DIAMETER clients and servers are based on OpenDIAMETER [9], an open-source implementation of the DIAMETER protocol. It must be noted that the clients are modified in order to accept and interpret incoming proprietary AAA requests, as well as package and send outgoing AAA responses.

Entry points to clusters accept incoming requests, and distribute them accordingly to cluster's computers. The entry point to server cluster is based on relay function of DIAMETER protocol and the entry point to client cluster is based on modified open-source load balancer Pen [10]. Entry points are made redundant in order to avoid bottlenecks, as well as achieving higher degree of reliability.

Implementation of high availability and reliability is facilitated with the use of OpenSAF [11]. OpenSAF is an open-source high availability middleware created.
according to Service Availability Forum's specifications [12]. We use it as a means for achieving redundancy of all computers in a cluster, distributed communication within a cluster, and application execution tracking and restoration.

B. Instrumentation and modeling

Instrumentation and modeling of the prototype are tightly related to one another. The model abstracts real-world complexity of the telephone exchange and the new non-legacy subsystem, while instrumentation provides real-world values for model's parameters.

In order to properly model and instrument the prototype, it is necessary to understand and abstract it's main functions. There are two main functions that the prototype executes - message processing and load balancing. Message processing includes creation of AAA request message on the telephone exchange, interpretation of that message on a client within clients cluster, creation of DIAMETER request message on the client, interpretation of DIAMETER request message on a server within servers cluster and creation of DIAMETER response message on the server. Furthermore, message processing also includes the return of the response messages through the same pathway used by the request messages.

The second function, load balancing, is executed on cluster's entry points when a request message arrives.

We model designated functions as common computing tasks, characterized by arrival rate and service rate of M/M/1 queue [13]. Arrival rate is based on the number and size of AAA messages that are sent from the telephone exchange. Service rate is calculated from two experimentally measured values. The first is maximum throughput of message processing and load balancing. The second is real-time value of processor load. As the load of processor increases, the maximum throughput for performing certain function decreases.

Besides message processing and load balancing, messages must be transferred through the link between AXE node and clients cluster and the link between clusters. Message transfer depends on the size of the message and the throughput of the link. It is important to note that both AAA and DIAMETER messages are small, with size significantly less than 1 KB. Furthermore, the link between AXE node and clients cluster, and the link between clusters, has very high throughput (1 GBit/s) and low-latency. Thus, for the most conditions, message transfer is very fast. So, message transfer has significantly smaller impact on prototype's performance than message processing and load balancing. Noted message and link characteristics enable us to model all message transfers as delay stations. Service rate of the delay station depends on current link throughput and size of the message that needs to be transferred.

Hence, we model the time of message processing on the clients and the servers, the time of load balancing on clusters' entry points, and the time of message transfer between AXE node, clients cluster and servers cluster. The sum of these times represents overall time spent for completing one authorization of an incoming SIP call.

IV. MEASUREMENTS

In order to verify basic concepts of black-box and gray-box instrumentation modeling, it is necessary to compare the results obtained by an experiment and the corresponding results obtained by a model performance prediction. We choose a simple scenario for the experiment and the model. The scenario follows conceptual architecture presented in previous section.

The selected scenario simulates a load on a small scale equipment. Incoming SIP calls are generated in the AXE node with maximum of 10 concurrent calls. Each call needs to be authorized in the DIAMETER non-legacy subsystem. After authorization the call is terminated and the AXE node generates a new SIP call. The experiment measures, and the model predicts, a number of SIP calls, in a 10 second time interval, whose authorization process was completed.

The scenario also explores what is happening to the number of completed SIP authorizations when the processing and link conditions in the non-legacy subsystem are less than ideal. Thus, on the one hand, the scenario is repeated for different overhead CPU loads of each computer of non-legacy system. Furthermore, on the other hand, the scenario is repeated for different decreased throughputs of network link that connects the non-legacy subsystem with the AXE node.

Deployment scenario for described experiments is shown on Figure 2. Computers are equipped with Intel...
In the paper we assess the usefulness of an introduction to basic modeling and performance prediction to a development process. In order to verify the usefulness, a prototype system was built. The prototype consists of the existing legacy system telephone exchange and the new non-legacy subsystem that adds additional, AAA functionality to the legacy system. The non-legacy system is built upon open-source solutions that enable rapid software development.

The method used for modeling and performance prediction is in accordance with the current research progress of the Q-ImPrESS project which we are a part of. We use M/M/1 queues and delay stations for modeling different parts of the prototype. The parameters of queues and delay stations are obtained by instrumenting the corresponding parts of the prototype.

The comparison between predicted values and experimental results reveals many similarities. Even in changing CPU and network link conditions, relative tendencies and absolute amounts of predicted and experimentally measured values highly correspond to one another. Thus, we conclude successful outcome of the method's basic verification.

In the future work we plan to devise more complex verification scenarios. This will be possible as the method and tools of the Q-ImPrESS project get more mature. The knowledge gained from this basic verification scenario gives us valuable experience for forthcoming experiments and tests.

**ACKNOWLEDGMENT**

This work was partially funded in the context of the Q-ImPRESS research project (FP7-215013) by the European Union under the Information and Communication Technologies priority of the Seventh Research Framework Programme.

**REFERENCES**


