Nirdizati: A Web-based Tool for Predictive Process Monitoring

Andrii Rozumnyi[1], Chiara Di Francescomarino[2], Chiara Ghidini[2], Fabrizio Maria Maggi[1], Ilya Verenich[3,1], Kerwin Jorbina[1], Marcello La Rosa[3], Marlon Dumas[1], and Simon Raboczi[3⋆]

[1] University of Tartu, Estonia
{andriiro,f.m.maggi,marlon.dumas}@ut.ee,kerwin.jorbina@gmail.com
[2] FBK IRST, Trento, Italy
dfmchiara@fbk.eu
[3] Queensland University of Technology, Australia
{ilya.verenich,m.larosa,simon.raboczi}@qut.edu.au

Abstract. In this paper, we present a prototype of a web-based application for predictive process monitoring that can be used by process workers and operational managers to predict the future development of a currently running process execution. The implemented solution, named *Nirdizati*, is a configurable full-stack web application that supports users in selecting the preferred prediction methods from a list of implemented algorithms and enables the continuous prediction of various measures of interest at runtime. The results of the predictions, as well as the real-time summary statistics about the process executions, are presented in a dashboard that offers multiple visualization options. The target audience of this demonstration includes process mining researchers as well as practitioners interested in exploring the potential of predictive process monitoring.

Keywords: Process Mining, Predictive Process Monitoring, Machine Learning

1 Introduction

Predictive Process Monitoring [**?**] is an emerging paradigm based on the continuous generation of predictions about the future values of user-specified measures of interest about a currently running process execution. In this paradigm, a user define the type of predictions she is interested in and a set of historical execution traces. Based on the analysis of these traces, the idea of predictive monitoring is to continuously provide the user with predictions and estimated values of the measure of interest. Such predictions generally depend both on: (i) the sequence of activities executed in a given case; and (ii) the values of data attributes after each activity execution in a case.

⋆ Author names are in alphabetical order

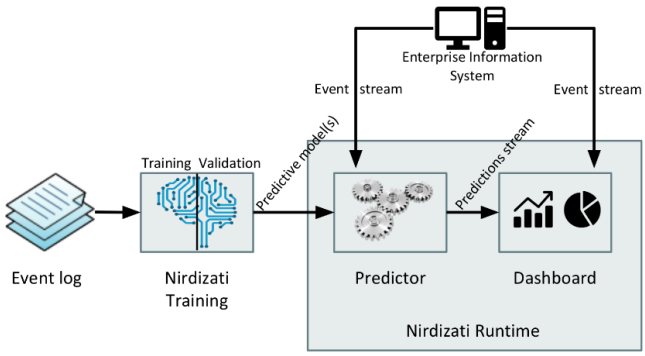

Fig. 1. A high-level overview of Nirdizati.

As an example, consider a doctor who needs to choose the most appropriate therapy for a patient. Historical data referring to patients with similar characteristics can be used to predict what therapy will be the most effective one and to advise the doctor accordingly. Meanwhile, in the context of a business process for managing loan applications, the applicant can be advised on the combinations of the loan amount and the length of loan that are the most likely to lead to acceptance of the application, given contextual information about the application and the personal data of the applicant (e.g., age, salary, etc.).

Several approaches have been proposed in the literature to deal with predictive process monitoring tasks. However, so far, the predictive monitoring approaches have largely remained in the academic domain and have not been widely applied in real-time scenarios, in which users require a continuous predictive support.

In this paper we present Nirdizati, a pioneering open-source Web-based predictive monitoring tool, which is able to fill this gap between research and practice, by providing business analysts with a highly configurable instrument for the configuration, generation and analysis of different predictive models; and end-users with continuous runtime predictions.

Nirdizati consists of two components: *Nirdizati Training* and *Nirdizati Runtime* (Figure **??**). Nirdizati Training takes as input a business process event log and produces one or more prediction models, which can then be deployed in Nirdizati Runtime. Once a model is deployed, Nirdizati Runtime listens to a stream of events related to a business process, and produces a stream of predictions. These predictions are then visualized in a continuously updated Web dashboard.

2 Nirdizati Training

Nirdizati Training is the component of Nirdizati that allows users to produce predictive models then used by Nirdizati Runtime for making predictions on

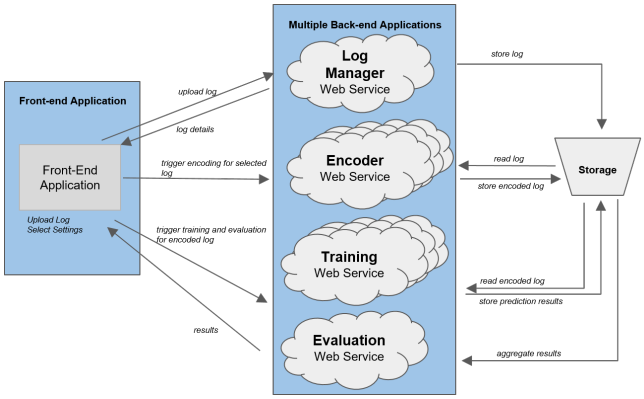

Fig. 2. A high-level overview of Nirdizati Training.

streams of events. It provides several algorithms for generating predictive models for different types of predictions and tailored to each specific dataset. For example, it is able to build predictive models for predicting remaining time, the next activity to be performed, whether a certain outcome will be achieved and even the overall workload per day. To this aim, the training component of Nirdizati relies on two phases: a training and a validation phase. In the former, one or more predictive models are built; in the latter, their suitability to the specific dataset is evaluated, so as to support the user in selecting the predictive model that ensures the best results.

Nirdizati Training is composed of a front-end application, which allows users to configure the predictive models and to evaluate the prediction results, and a back-end application responsible of the actual training and validation. The back-end application is in turn composed of the four submodules shown in Figure **??**. A Log Manager is in charge of managing the logs. Uploading and retrieving the logs are the basic operations of this module. The Encoder is responsible of parsing the logs, labeling them according to the desired type of prediction, and preparing the data for the training phase. In the Training submodule, the encoded data are split into training and validation set used for evaluation purposes, and the predictive model(s) is(are) built from the training data. Finally, the Evaluation submodule tests the validation data against the model(s) created to get accuracy measures with respect to the groundtruth, available from the complete traces in the validation data. The back-end application comes with a storage module for saving the uploaded logs and the built predictive models.

3 Nirdizati Runtime

Figure **??** illustrates the flow of data through the proposed system. A workflow management system (WFMS) continuously registers tasks performed in the organization. This serves as an input to our system, in the form of a stream of

events. Next, the produced stream is consumed by a web server which is responsible for storing data and further information processing. Web server, in turn, communicates with a predictive engine which applies pre-trained models for a business process and returns all the results of calculations. After web server receives the outputs from each predictive model, it writes them to a database and updates corresponding clients. From the user's side, the results are visualized via a dashboard-based interface which is capable of sending queries requesting various visualization options.

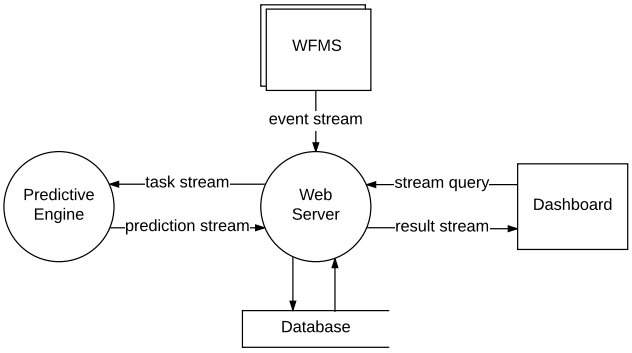

Fig. 3. High-level data flow diagram in a Yourdon-Coad notation

Based on this data flow diagram, we devise the main parts and high-level components of the system (Figure **??**):

- streaming platform
- web server with load balancer and internal components
- predictive engine
- client-side user interfaces

Process workers and operational managers typical users of such system can set the process performance targets and subscribe to a stream of warnings and alerts generated whenever these targets are predicted to be violated. Thus, process workers will be capable of making more informed, data-driven decisions to get a better control of the process execution. This is especially beneficial for processes where process workers have more leeway to make corrective actions, for example, in a lead management process.

4 Conclusion

The developed solution, named *Nirdizati*, is a configurable full-stack web application that supports users in selecting the preferred prediction method from the

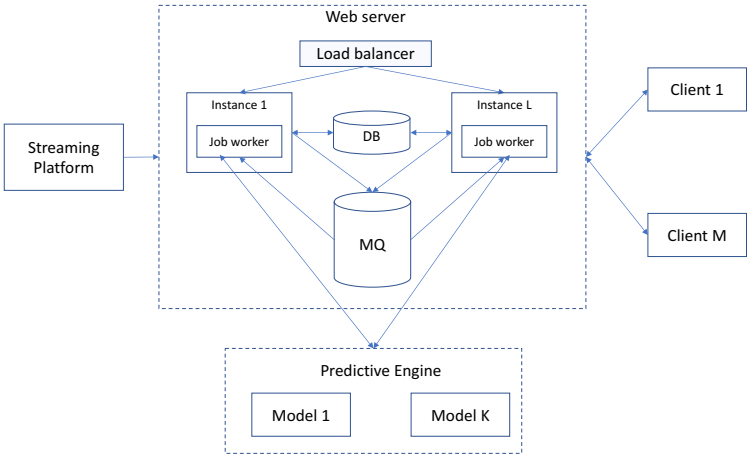

Fig. 4. Proposed system design architecture

list of implemented methods and enables the continuous prediction of various performance indicators at runtime. The results of the predictions, as well as the real-time summary statistics about the process execution, are presented in a dashboard that offers multiple visualization options.

The source code has been released as open source software under the Lesser GNU Public License (L-GPL) at `http://github.com/nirdizati`. The developed engine has been deployed on the server belonging to the Institute of Computer Science and can be accessed at `http://nirdizati.com`.