User Interaction with a Technology Evolution Model to Select Most Viable Enterprise Information Systems

Andrzej M.J.Skulimowski[1] [0000-0003-0646-2858], Inez Badecka[1,2]
[1]AGH University of Science & Technology, Department of Automatic Control and Robotics, Kraków, Poland
[2]International Centre for Decision Sciences and Forecasting, Progress & Business Foundation, Kraków, Poland
{ams, inezbad} (at) agh.edu.pl

Abstract— **One of alternative ways to achieve a competitive advantage by an enterprise in the ICT area is to select the most prospective information system (IS) as a seed technology. The latter is defined as a first technology from a given area implemented in an enterprise, that determines or strongly influences the choice of further technologies from the same area. Seed ICT selection is particularly relevant for enterprises whose core business activity relies on online service provision. In this paper we present a user-requirement-based interaction scheme of an application that models the evolution of enterprise ISs, taking into account the production patterns of software releases, the functionalities offered, the users' community growth trends, and other factors. The evolution is modeled by hybrid discrete-time/discrete-event systems, and multi-models built with them. The technology choice is supported by multicriteria ranking algorithms that apply IS innovativeness forecasts. Moreover, this evolution model of enterprise IS can handle different uncertainties and cyber security issues. We present the use cases and scenarios, model management, and the knowledge base that stores time series, IS characteristics and forecasts. The use of this application is illustrated by a real-life example of three popular open-source CMSs. We conclude that an informed choice of the first IS and the corresponding enterprise architecture as a seed technology is particularly relevant for start-ups and SMEs. An extended variant of the above application may also provide decision support to developer teams seeking software evolution models to discover and apply the best-possible technology development and market strategies.**

Keywords—Software evolution, Online services provision, Information systems, Interface design, Multimodels, Versioning

I. INTRODUCTION

Innovation policy related to the deployment of different information and communication technologies (ICT) in enterprises is an important problem from the perspective of the software market and online services provision. In large enterprises, the scheduling of ICT innovation deployment has been extensively studied by many researchers, cf. e.g, [6]. The markets of key technologies relevant to major corporations as well as the evolution of consumer ICT are well-known to market experts in contrast to the ICT dedicated to small and medium-sized enterprises (SMEs) that offer online services.

An accelerated speed of evolution of information technologies, faster than all other technology areas, brings more progress and price decline to consumers. On the other hand, the delivery of numerous software releases, new services and functionalities offered to customers, growing demand for support, system compatibility and cyber security issues, along with the growing risk and value of losses in the case of changing technology are factors that make corporate user decisions concerning ICT selection increasingly harder. Another research challenge involves forecasting the development of ICT, specifically the online service provision systems which in SMEs are frequently integrated with enterprise information systems (EISs). Forecasts may assist the enterprise CIOs in making optimal decisions regarding the choice of the most prospective EIS. Additionally, potential corporate ICT users are uncertain about the future of information systems provision systems considered for selection as increasing competition produces the threats of financial default by developer companies or a cease of production. Investments in human resources (HR), the service provision by the system owner, and the overall compatible enterprise software ecosystem are jeopardized by such unfavorable development scenarios.

This paper is concerned with the information system evolution prospects from the corporate (predominantly SME) user point of view. This is why all fea¬tures, and modules and are seen as user-relevant functionalities or services, while their software architecture, re-use opportuni¬ties and development resources are assumed unknown to the user.

A. Related work

Among other factors, release generation by software providers is of relevance when modeling ICT evolution. For example, in [7], a prediction model based on the analysis of past versions of 10 open-source projects was applied to forecasting trends in the evolution of networks representing Java systems. The trends were extrapolated, taking into account object-oriented software design rules and additional model components, such as the developer collaboration network dynamics and disruption monitoring. Another kind of software evolution model, is investigated in [10], where the releases are user-driven rather than competition-driven. The users of an open source software could suggest improvements as well as removal of software bugs. These suggestions were regarded as the main input in designing the Eclipse (next release). Various seasonal ARIMA models were used to forecast the inflow of change requests in this software project, and thus the generation of releases. Further software evolution models use feature structure [2] or other software product lines (SPL) paradigms [13]. This kind of evolution analysis is relevant from the software developers'

978-1-7281-5411-4/19/$31.00 ©2019 IEEE
DOI 10.1109/SOCA.2019.00030

IEEE computer society

perspective, as they are well acquainted with the best code structure. Software evolution models based on an automated analysis of versions have been proposed in [9]. The models of functionality dynamics, as seen by the users, have been constructed and analyzed in [16] and [17] for the most popular open-source content management systems (CMSs). These models use systems of quasi-linear stochastic equations with coefficients estimated with vector autoregression (VAR). In the above cited paper [16], we introduced the notion of the *seed technology*, which can be defined as a first technology from a given area implemented in an enterprise which strongly influences the choice of further technologies from the same area. The mechanisms of the above influence are mostly driven by compatibility and human resources, although other factors such as supplier selection may also play an important role. The relevance of seed technologies in the ICT area have been also pointed out in [15], referring to the choice of a CMS in SMEs. Seed technology selection is particularly vulnerable to the aforementioned threats and requires penetrative decision analysis which takes into account all available supporting information.

B. An overview and the structure of this paper

This paper presents a bottom-up design process of an application that supports decisions related to the choice of a seed information technology for an EIS. It implements empirical observations of best practices of real-life decision processes of the same type but performed without systematic computer support. As additional information supports the above-mentioned decisions, we propose parametric models of software evolution in various development scenarios and the user community development prospects. For this application, we distinguish two general use cases related to target application users. The first group includes users and potential users of the ISs under study. They are mostly interested in the technical advancement of the application and its user-friendliness because the application will support them in the choice of an EIS. The second group of users is involved in software development for such systems.

Here, we are only concerned with a detailed study of the first use case. In sec. II we describe the functionalities and software architecture of the application that – based on earlier best decision practices – was supposed to cover the overall decision process in the first above general use case. Then, in sec. III, we present the use case and scenarios, relating them to the structure and uses of the application interface. Since each of these use cases does not need to cover the entire technology analysis and decision process, we will term them sub use cases. In subsection III.B we refer to the VAR models and endow them with further factors and analysis methods. Sec. IV explains how the sub use cases are combined to support the user's decision on an ICT selection based on its viability. We use the standard UML notation to describe the system's architecture and the interface structure, cf. e.g. [1]. In sec. V we present an application example referring to the popular open source CMS evolution models [16],[17]. This example can be regarded as a validation of the use case 1 workflows and modeling algorithms on test data. The prospective user can select the system best fitting the predefined selection criteria. In the conclusive section VI we summarize the benefits of using this kind of decision support systems (DSS) and discuss its further applications to solve online service providers' competition strategy choice.

II. FUNCTIONALITIES OF THE IS EVOLUTION MODELING APPLICATION

The goal of the evolution modeling application, termed VF Manager, is to generate forecasts for information systems development based upon the dates of introducing new versions (V) and functionalities (F).

For this end, the numerical calculation module processes historical data loaded from the system history file or entered by the user. Based on selected observation data, the coefficients of models describing the future technological innovations in all systems will be estimated. Forecasts until a user-defined time horizon or until a given number of simulation iterations is reached can be generated with each model. The application calculates basic statistics such as the mean number of software innovations, average technological level growth and confidence intervals for them at a given level α. VF Manager allows its users to generate forecasts with several models and pairwise compare them in the same interface window.

As already mentioned in sec. I, VF Manager may be used in two main use cases. In the use case analyzed in this paper, a potential system user is interested in getting an efficient and viable services provision tool. In the second use case, a developers' team seeks model-based decision support allowing them to discover and apply best-possible technology development and market strategies. This group will use the application to establish an optimal strategy for the development of the system while taking into account the competitors' offers as well as attempting to find the development path that optimizes the perceived attractiveness of the system in the eyes of users.

Both above use cases are based on a similar theoretical and modeling background, yet their decision-theoretic context and goals are different, especially considering that developers cannot select the system they've already developed. To comply with this use case, an important criterion of the application design is its accessibility to users with limited IT skills. The result of this is a restriction on the user interface complexity. The interaction with VF Manager starts in the main window of the interface. Visualizations and comparisons of forecasts are performed in additional subordinate windows. No data bind them with parent windows, therefore, changes in a subordinated window only affect local representation of data. The interface is organized into several separate functional areas. This version of VF Manager is endowed with the following areas:

1. Data management.
2. Model management- interactive support of the discrete-time / discrete-event system model construction, generating a set of models that meet the given conditions, and multi-model construction, cf. [5].
3. Forecast management- operations on the results of forecasts obtained using selected models. This area contains sub-modules (sub-areas):
   - Simulation: designing simulation experiments for a selected model or multi-model and calculating statistical characteristics of simulation results.
   - Optimization and rankings: includes computing nondominated model parameters with respect to

information [12] and fitting criteria, optimal multi-model weighting coefficients.

A scheme of the interface containing the above functional areas is presented in Fig. 1. Its detailed characteristics are provided in section III. Optimization and ranking area constitutes a separate module based on a data interchange (cf. section IV).

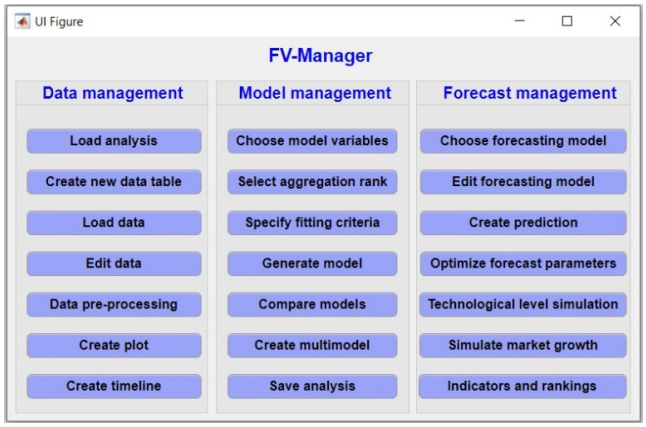


Fig. 1. The main screen of the VF Manager interface.

Specifically, the present version of VF Manager is dedicated to solving the system or technology selection problems, where the number of selection options – the competing information systems or other ICTs– is usually small, i.e. in the range of 3-7. This is why VF Manager does not offer a full variety of multicriteria analysis, ranking and optimization procedures. The user may define quality indicators to compare and rank the systems, and choose a scoring functions to be used in rankings. The interface of VF Manager provides merely links to external procedures and transfers input data in the standard format aligned to the requirements of multicriteria analysis methods available. The 'Optimization' sub-area is planned in the future versions of this application. It will deliver more options to the user, who will be able to choose sophisticated methods of multicriteria analysis, for example reference sets. More details on solving the system selection problem are provided in sec. IV.

III. Use Cases and Scenarios

As stated above, the companies who select a technology for a service provision system focus on using the VF Manager to get an insight into the future of systems development, such as the prospective technological advancement, ease of use, etc. The knowledge gained when interacting with the evolution modeling application can be used in further decision analysis to help users in choosing the information system or technology. For this group a software evolution model additionally supports corporate learning based on predicted results of EIS competition. These uses may be particularly beneficial when selecting an information system as a seed technology. In accordance with the commonly accepted convention [3], using the system for a specific purpose will be termed the use case. Each use case will be associated with a specific functional area in the interface [8]. Uses of this area related to the attainment of specific goals will be referred to as sub use cases. A set of use cases of the same application grouped by similar goals, concatenated uses of different application features or modules, or clusters defined by other relations will be termed general use cases (GUC). A GUC does not usually take into account all possible system uses. It focuses on use cases important for a specific group of users. This convention usually implies two or more GUCs. The concrete application uses within the GUCs will be referred to as scenarios [20].

The use case diagram of VF Manager is presented in Fig. 2 below.

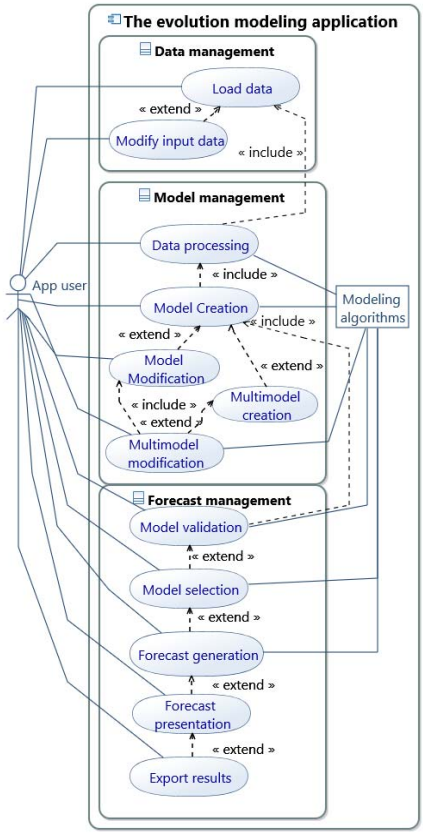


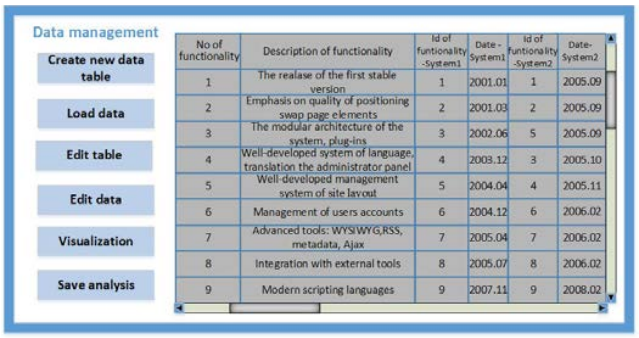
Fig. 2. Application sub use cases for the General Use Case I

Since in this paper we will only present the first of the above GUCs, we will refer, without ambiguity, to sub use cases just as use cases. Let us note that most of the sub use cases coincide in both GUCs, and the main differences are intrinsic at the final recommendation stage only.

The data processed in the "Data Management" area and fed to analytic procedures in other areas may contain the following components:

- information about versions and functionalities of *N* information systems co-analyzed,
- information on the number of each system users at the end of each historic period,
- information on the expenses required to implement new versions of each system during previous periods,
- information on the technological level achieved by each system at the end of each historic period.

Fig. 3 presents a sample window of this interface area.



No of functionality	Description of functionality	Id of funtionality -System1	Date - System1	Id of funtionality -System2	Date- System2
1	The realase of the first stable version	1	2001.01	1	2005.09
2	Emphasis on quality of positioning swap page elements	2	2001.03	2	2005.09
3	The modular architecture of the system, plug-ins	3	2002.06	5	2005.09
4	Well-developed system of language, translation the administrator panel	4	2003.12	3	2005.10
5	Well-developed management system of site layout	5	2004.04	4	2005.11
6	Management of users accounts	6	2004.12	6	2006.02
7	Advanced tools: WYSIWYG,RSS, metadata, Ajax	7	2005.04	7	2006.02
8	Integration with external tools	8	2005.07	8	2006.02
9	Modern scripting languages	9	2007.11	9	2008.02

Fig. 3. A sample view of data management area in the VF Manager interface

Below we provide further details on VF Manager sub use cases with references to the functional sections of the interface.

A. *Use cases and procedures related to model management*

Fig. 4 presents use cases related to the "Model management" area. The procedures corresponding to this area build one or more forecasting models for a given dataset. The user has the possibility to co-create the model semi-automatically, with the support of the appropriate application guiding module. After generating the model, the user can add it to a model group or just save and retain it for further analysis.

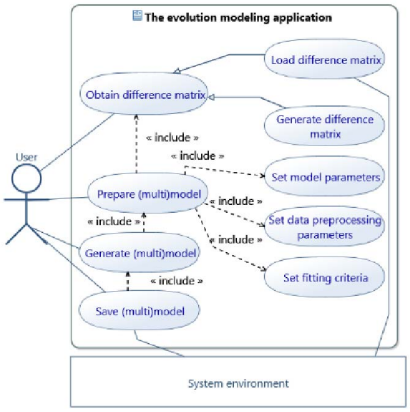


Fig. 4. The application use cases in model management area.

The models created in VF Manager will simulate the generation of future innovations in the information systems under study. VF Manager can estimate the coefficients of quasi-linear vector autoregression models of the form:

$$X_t = A_1 X_{t-1} + A_2 X_{t-2} + \cdots + A_p X_{t-p} + a + \varepsilon_t \qquad (1)$$

where:

- $X_t=(x_1(t),x_2(t),\ldots,x_n(t))^T$ is an n-dimensional vector time series, which coordinates are state variables $x_i(t)$, for $t=p,p+1,\ldots,T$, and $i=1,\ldots,n$,
- A_i, are $(n\times n)$ matrices with estimated model coefficients, $i=1,\ldots p$,
- $a=(a_1, a_2,\ldots,a_n)$ is a vector of trend drift,
- $\varepsilon_t \in \mathbf{R}^n$, is a vector of normally distributed random disturbance terms satisfying: $\varepsilon_{it} \sim N(0,\delta_{it})$, with $\delta_t<\delta_0$, for $i=1,\ldots,n$.

The state variables x_i can be defined as either time lapses between the market delivery moments of subsequent functionalities or versions, or frequencies of releases during a certain period of time. Explanatory variables for x_i are either historical values of x_i and other state variables or average values on them calculated on fixed-length time intervals. The latter terms are, in fact, nonlinear because the model (1) is asynchronous, i.e. the real time lapse between i and i+1 may vary. The autoregression rank need not be equal for all systems included in the model. Additionally, models may contain assessments of the technological level achieved $d_i(t)>0$ and other time-varying parameters that characterize emerging innovations in information systems. Periodic sales value $s_i(t)$ or a number of users $c_i(t)$ (the latter in case of open-source systems) can be considered as output variables that serve to calibrate model parameters, while at the same time serving as one of the selection criteria (cf. the next section). Finally, users can combine one or more models of type (1) with an output equation

$$Y(t)=f(X_t,X_{t-1},\ldots,X_{t-k})+\eta_t \qquad (2)$$

where f is the estimated regression, η_t is a random observation error, $y(t):=(y_1(t),\ldots, y_{n1}(t))$, $n1\le n$, and $y_i(t)$ is either $c_i(t)$ or $s_i(t)$.

The evolution model construction includes selecting model variables, estimating the coefficients, and calculating their basic *ex-ante* statistics. Alternatively, a user can construct a certain number of different models and combine them into a multi-model. Statistical inference based on a set of models (multi-model inference) has been described in [5]. Additional properties of multi-models and the algorithms for building them are presented e.g. in [19]. The multi-models used by VF Manager aggregate models linearly, with weighting coefficients, learned with an optimization procedure to ensure adaptive best fitting during the learning period. The number of models included in a multi-model is controlled with Akaike or Bayesian information criteria [12]. Users choose a trade-off between the goodness of fit and one of above information criteria. Usually, a multi-model will be constructed if none of the previously created models yields satisfactory forecasts. In such a case, the creation of a multi-model is preceded by the construction of at least two different evolution models for the same or different sets of competing systems. A multi-model building algorithm is presented below.

Algorithm 1.

Input data: Historical data on the systems releases and or functionalities transformed into a time series representation suitable for modeling purposes. Time series are stored in the matrices $A_i:=[a_{ij}]$, $B_i:=[b_{ij}]$, $C_i:=[c_{ij}]$, $D_i:=[d_{ij}]$, where:

a_{ij} – time intervals between the dates of the introduction two consecutive functionalities / versions within the same system calculated with a given accuracy of k months,
b_{ij} – time intervals between the dates of introducing the same functionalities in different systems,
c_{ij} – the number of users of each system at the end of each time period j (which can be interpolated by the application)
d_{ij} – a technological level assessment of each system.

Step 1. User selects a data pre-processing procedure, then the variables for the model. The pre-processing includes a method

of averaging state variables, either r_i recent ones or a variable number of them from a fixed-length sliding time window.

Step 2. User defines the scope of statistical analysis, methods to be used, and the required statistical parameters of the model. We assume that the time interval of the subsequent *i*-th system release depends linearly on the p_i-1 previous time intervals between releases of this system and on the frequency of releases of other systems. The model parameters to be determined are:

- The autoregression order p_i which can be chosen automatically by the system with VARIMA algorithms or can be defined by the user in the range from 1 to p.
- The order k_{ij} of inter-regressions, i.e. the maximum number of previous periods when the coefficients $a_{ij}(p)$ of matrices A_p in (1) may be non-zero. The inter-regression variables are nonlinear and characterize the dependence of time needed to generate the next release of the *i*-th system on the number of hitherto releases of other systems in k_{ij} former periods.

Step 3. Choose the stationarity, best-fit, and information criteria to evaluate the model. The available parameters are:

- The score and *p*-value of stationarity and unit root tests (Dickey-Fuller, KPSS).
- Sub-period of the observation period to calculate the mean-squares error.
- Bayesian Information Criterion (BIC).
- Akaike Information Criterion (AIC).

Step 4. Choose an initial trade-off coefficient between the fitting and information criteria and the no. of models to generate.

Step 5. Generate and evaluate the models according to **Step 4** and the settings selected previously. Display model coefficients and the fitted processes.

Step 6. Generate forecasts according to a separate model iteration procedure. Display forecasts received with the models ensuring (a) the smallest ex-ante error and (b) best information criterion value and the highest ex-ante error acceptable.

If (a) is not acceptable **then** mark the flag *F* and perform

Step 7. Generate multi-models from the models calculated so far listed by (i) assembling different parts of various models or (ii) by a linear combination of models. The case (i) can be accomplished by combining coefficients and picking rules from different models.
Proceed to **Step 6** to find multi-models corresponding to the criteria values (a), (b) with an optimization procedure.

Step 8. **If** no acceptable forecast could be generated the user can modify the retrieved rules or add his/her own rules describing possible behaviors of observed systems.

If the flag *F* is marked at most once **Proceed to Step 6**
Else Repeat the **Steps 1-6** until an appropriate model is found or the modeling resources are exhausted.

Else proceed with accepted forecasts to the ranking procedure to elicit the most prospective information system. ■

The use case „Choice of model variables" (cf. Fig. 1) is included in the „Modify models parameters" process in model management area as „Request to change model parameters". Furthermore, the use case "Data pre-processing" allows the user to average over the events count. The menu contains the following values: 1 – no averaging,…, *q*-averaging over *q* values, *t*-averaging on sliding time interval of length *t*, "Auto" - automatic choice of the averaging period. After selecting the last option, one can select one or more criteria from among: maximum *p*-value, minimal error value, the minimal value of the AIC and BIC criteria. In case of averaging over a time interval, the user chooses its length as a number of months. A simple model building example is presented in sec. V.

Model building procedures are capable of handling a situation, where there is insufficient or no data to define the coefficients. This may happen if, in the coefficient learning period, one or more of the system development teams operates autonomously, without taking into account the state of other systems. However, this strategy may change during the forecasting period. The model fitting procedure then seeks analogous relations between other systems and inserts coefficients generated with the maximum likelihood principle into the forecasting model. Such situation has been encountered with the open-source CMSs described in [17], where Drupal was the unique innovation leader during the entire observation period.

B. Use cases related to forecast management

The "Forecast management" area controls procedures using the evolution models to generate forecasts and simulation results. It gives the option to adjust previously generated and saved evolution models, and to launch the forecasts. In addition, the existing models can be enhanced with decision rules concerning the choice of technological features in the next release of a modeled IS. The identified or assumed decision rules can fill the gaps in the models. They might also be used in the second GUC as developer-driven controls that can be optimized to achieve a competitive advantage by the corresponding IS.

With the forecast management procedures the users can associate multiple variants of forecasts with models, re-use and merge other forecasts and their underlying models. An important functionality offered in this area is forecast visualization and comparison. Completed calculations or partial results can be saved to be available for further analysis without a need for reprocessing.

The order the same functionalities appear in different systems usually varies from system to system, so the next service or functionality of a modeled system does not need to be that recently introduced by another system. Hence, it is necessary to identify functionality inclusion rules in new releases. The forecasts combined with rules can be generated with a hybrid approach: the model (1)-(2) or a suitable multi-model calculates the next-release dates, while the supplementary decision rule determines the functionality to be implemented next.

The use case „Choose forecasting parameters" determines how the forecasting process is executed, cf. Fig. 5 below. After pressing the "Create prediction" button, VF Manager displays a new window with the list of available models. The user chooses the models and time range, and/or the number of simulation iterations. The overall actions and activities of the "Forecast management" area are presented in Fig. 6 (next page).

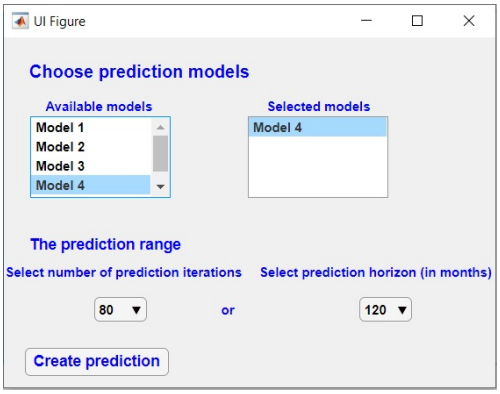


Fig. 5. The interface window presenting forecasting model selection.

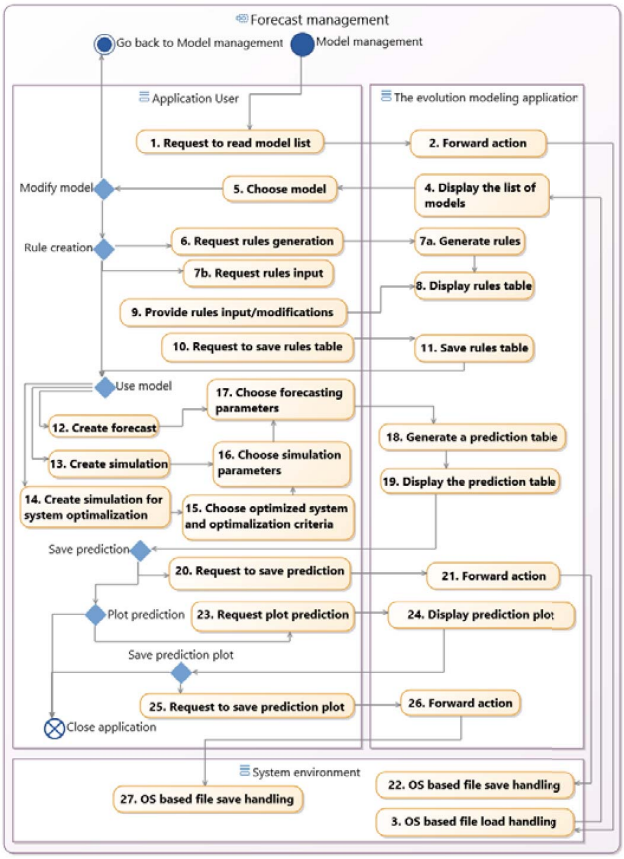


Fig. 6. The activities and actions diagram of forecast management area

IV. Decision Analysis and Support Procedures

The aim of the use case "Generate ranking" is to create a ranking of information systems based on forecast results and performance indicators of each system. This use case complements the above-presented uses of VF Manager, but it is not fully integrated with this application, giving the user the freedom to define various optimality criteria, preference models and ranking algorithms. For the open source systems analyzed in [17], we admitted the following assessment criteria:

i) the expected number of functionalities in a given forecast period $[t_0,T]$, or
ii) the forecasted level of technological excellence to be reached at T, where T is the forecasting horizon, and
iii) popularity expressed by the expected number of users of the given system at T.

The values of the criterion (iii) are extrapolated from data retrieved from [4], taking into account a system dynamics dependence on the technological excellence level.

The user can select a multicriteria outranking method to aggregate the above criteria. The currently available methods are simple scoring function with positive weights, the multiple reference point, and reference set methods. However, the user can select another ranking method as he/she sees fit. The ranking is intended to support the user's decision by an obvious rule that the top-ranked system is to be chosen for implementation. This decision, when made, is communicated to the VF Manager, which will store the choice and use it in a learning scheme for consecutive modeling sessions with the same or other users. The ranking procedure runs in the following way:

Algorithm 2.

Step 1. User selects from the pick list the quality indicators $\{F_1,...,F_n\}$ to compare and rank the systems, or defines a new indicator as a function of forecasted variables available and include it in the set of criteria.

Step 2. The systems to compare, $S:=\{S_1,...,S_k\}$, are picked from a list of available system development forecasts (usually all are selected, but users can exclude some of the systems based on their forecasted basic characteristics only).

Step 3. User exports forecast characteristics to be used in calculating the ranking from criteria values with 'Export data'.

Step 4. Forecasts of external variables $h_1,...,h_p$ (not included in the forecasting model, e.g. sales, macroeconomic variables) are imported or inserted from the console by the user.

Step 5. User defines functions (weighted sums and quotients) that combine the external variables with forecasts of systems' characteristics to yield the final ranking criteria $\{F_1^*,..., F_n^*\}$ and calculates their values $F_i^*(S_j)$ for each system to be ranked.

Step 6. The criteria values characterizing the selected systems are displayed in a 2D or 3D coordinate system, where the axes correspond to criteria. The set of nondominated systems P(S) is calculated and marked in the diagram.

If P*(S)* consists of one element or the user prefers to choose the system based on the graphical representation of criteria, only **Stop**
Else Proceed to Step 6

Step 7. User defines the coefficients of the scoring function or links an external outranking algorithm.

Step 8. **If** the best ranked element conforms to the non-disclosed preferences of the user **Stop.**

Else Repeat Step 6 until a satisfactory solution is reached or the ranking resources are exhausted. ■

Among the multicriteria analysis methods, the user can choose one of the popular outranking algorithms, for example, reference points or sets, future consequences or trade-off convergence analysis.

The ranking module recommends the following criteria to start with initial weighting coefficient values equal to 1:

- The number of functionalities of the system concerned reached at the end of forecasting period T.
- The number of functionalities of the system concerned reached at an intermediate planning horizon $\tau<T$.
- The forecasted technological level reached at T combined with its growth ratio.
- The forecasted technological level combined with its growth ratio an intermediate planning horizon $\tau<T$.
- The forecasted mean frequency of essential releases.
- A linear combination of the forecasted number of users and the average user community growth ratio.

The user can pick several criteria from the above list and use them in the outranking algorithm. The criteria values are first standardized on the interval [0,1]. This justifies the recommended initial scoring coefficients values equal to 1. As outlined in sec. II, a variety of procedures allowing users to rank the systems according to the given criteria and scoring functions may be imported from the separate Optimization and ranking functional area and used interactively as a group DSS. A ranking example is provided in the next section.

V. AN ILLUSTRATIVE EXAMPLE

This section presents an implementation example of applying use cases from all of the above functional areas. As systems to be compared serve three popular open-source CMSs: Wordpress, Drupal and Joomla! CMS selection is a strategic decision for an enterprise [11] and a CMS can be regarded as a seed technology [17]. The input information such as description of functionalities and the corresponding release dates comes from the websites of each CMS and is verified and edited in a form suitable for model building. The information about the CMS market comes from [4]. The market shares for the above three systems at the end of 2016 were respectively 51%, 9%, and 11%. The visualization procedure in the data management area gives a user the means to visualize historical data as a table, a plot or a timeline, cf. [17]. Users can also visualize data related to the market and technological level that can be plotted in 2D or 3D. The model generated for the three open-source CMSs with its statistics is presented in the VF Manager interface as shown in Fig. 7.

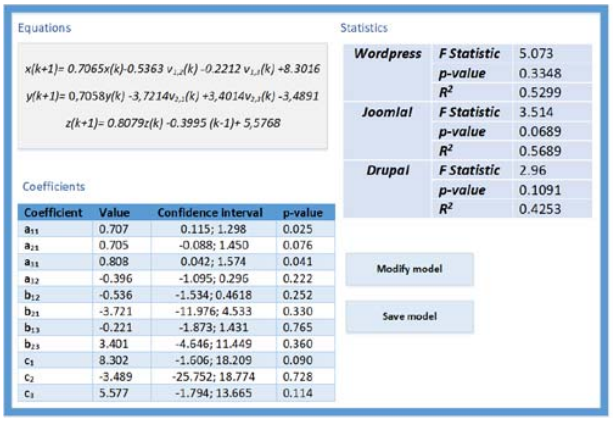


Fig. 7. An interface window presenting the VAR model parameters

The variables $x(k)$, $y(k)$, and $z(k)$ in the upper left window denote time intervals between next (k-th) essential system improvement, while $v_{ij}(k)$ for i=1,2 j=1,2,3, $i\neq j$ denote the dependence of the next functionality of the i-th system on the average frequencies of introducing new functionalities by of the j-th system during a given sliding time window [17]. The statistical significance of the model presented in Fig. 7 was confirmed with the Fisher-Snedecor [18] and goodness of fit tests with the $\mathbf{R}^2$ determination coefficient. The confidence intervals for the model's coefficients at p-value 0.05 were suitable to construct a meaningful ranking [16],[17]. Following [17], the general model eqs. can be presented as:

$$x_i(k+1)=a_{i,1}x_i(k)+a_{i,2}x_i(k-1)+\ldots+a_{in}x_i(k-n+1)+b_{i,1}v_{i,1}(k) +b_{i,2}v_{i,2}(k)+\ldots+b_{i,i-1}v_{i,i-1}(k)+b_{i,i+1}v_{i,i+1}(k)+\ldots+b_{i,n}v_{i,n}(k)+c_i \quad (3)$$

where $x_i(k)$ are time intervals between next (k-th) essential system improvement, a_{ij} and b_{ij} are model coefficients estimated by the VF Manager, and $v_{ij}(k)$ are as above.

With the procedures from the forecast management area, the application generates forecasts and their plots. An example of a data record on predicted values at a given forecasting horizon is presented in Table I.

TABLE I. AN EXAMPLE OF A DATA RECORD OF PREDICTED CMSS VALUES

Wordpress		**Drupal**		**Joomla**	
Forecast horizon	***Number of innovations***	***Forecast horizon***	***Number of innovations***	***Forecast horizon***	***Number of innovations***
2025-05	35	2025-07	35	2025-03	32

After generating the forecasts, the user can proceed to creating a ranking of forecasts and selecting the most suitable system. An implementation proposal of this is presented in Fig. 8. After clicking the button "Select systems and criteria", the application displays a new window with the list of available options. The user chooses criteria and systems to rank, then enters the forecast horizon which is common for all systems.

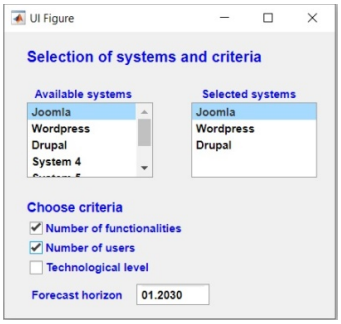


Fig. 8. Systems and criteria used to IS rankings.

Fig. 9 presents the generated ranking. The user chooses a ranking method from the list. The name of the selected method is displayed in the field "Selected method". The user can also define and add new method to the list using the button "Add new ranking method" which attaches a new Matlab script in a prescribed format.. The button "View ranking" triggers the ranking calculation. The best system is displayed in the highest position in the ranking. The "Save ranking" button allows the user to save the ranking results and her/his selection.

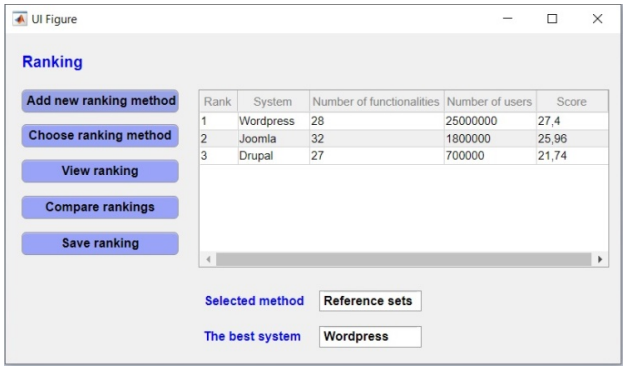


Fig. 9. Ranking results. The numbers of system users are retrieved from [4].

VI. CONCLUSIONS

The application presented in this paper proposes a new method of software evolution modeling based on the innovation delivery activity of software development teams. Real-life experiments showed that the number and frequency of software releases is a relevant source of historical data to be used for forecasting, which turned out to be particularly useful for open-source information systems. However, for commercial software, the version history can only define auxiliary variables that do not provide a complete description of the system evolution as the market placement strategy can play a fundamental role. However, a preliminary analysis of release data points out that the evolution of commercial systems basing strongly on research competition, such as medical DSS, may also be adequately described by the models of type (1)-(2). In addition, users or analysts must filter the software release history or roadmap to eliminate versions produced only to fix bugs or provide minor updates. Therefore, the principal role in building forecasting models is played by system functionalities that can also be easily transformed to define the technological advancement level of the system and merged with expert assessments. Therefore, modeling the functionality evolution requires domain knowledge to identify relevant innovations in software releases and match them with equivalent functionalities in descriptions of other systems included in the model. Functionality based evolution model building is even more complex when the functionality order varies for different systems and more than one functionality can be included in one software release. Despite all of the above modeling difficulties, the approach shown here proved useful in selecting an open source CMS as a seed technology for SMEs ([15], Ch.5) that used different criteria to assess SMEs needs, from technological excellence to ease of use.

There are further research prospects for the development of the application shown in this paper. Firstly, this version of VF Manager is dedicated to solving system or technology selection problems and does not offer a great deal of specialized multi-criteria analysis and optimization procedures. This application has been implemented in Matlab 2018b as a prototype research version. Further optimization procedures, such as optimal strategy planning for the developer teams, are planned as parts of the next version of VF Manager. It will feature an application demonstrator, which is intended to be made available online as a component of the technological forecasting area of foresight support systems [14].

REFERENCES

[1] D. Alur, D. Malks, and J. Crupi, "Core J2EE Patterns: Best Practices and Design Strategies", Prentice Hall/Sun Press, 2001.

[2] D. Benavides, S. Segura, and A. Ruiz-Cortés, "Automated analysis of feature models 20 years later", Information Systems, vol. 35(6), 2010, pp. 615–636.

[3] N. Bolloju, and S.X. Sun, "Exploiting the Complementary Relationship between Use Case Models and Activity Diagrams for Developing Quality Requirements Specifications", In: I.Y. Song et al. (eds), Advances in Conceptual Modeling – Challenges and Opportunities, Lecture Notes in Computer Science, vol 5232, Springer, Berlin, 2008, pp.144-153.

[4] Builtwith website (web technology lookup), trends.builtwith.com/cms. Accessed 09/2019.

[5] K.P. Burnham, and D.R. Anderson, "Model Selection and Multimodel Inference: A Practical Information-Theoretic Approach", Springer, 2002.

[6] J. Cardoso, R.P. Bostrom, and A. Sheth, "Workflow Management Systems and ERP Systems: Differences, Commonalities, and Applications, Information Technology Management, vol. 5, 2004, pp.319–338.

[7] T. Chaikalis, and A. Chatzigeorgiou, "Forecasting Java Software Evolution Trends Employing Network Models", IEEE Transactions On Software Engineering, vol. 41(6), 2015, pp.582-602.

[8] M. El-Attar, and J. Miller,"Constructing high quality use case models: a systematic review of current practices", Requirements Eng, vol. 17(3), 2012, pp. 187-201.

[9] G. Ghezzi, and H.C. Gall, "A framework for semi-automated software evolution analysis composition", Autom. Softw. Eng., vol. 20, 2013, pp. 463–496, DOI 10.1007/s10515-013-0125-z.

[10] M.Goulão, N. Fonte, M. Wermelinger, and F. Brito e Abreu, "Software Evolution Prediction Using Seasonal Time Analysis: A Comparative Study", Proceedings of the 2012 16th European Conference on Software Maintenance and Reengineering, March 27-30, 2012, pp.213-222.

[11] P. Hallikainen, H. Kivijarvi, and K. Nurmimaki "Evaluating strategic IT investments: an assessment of investment alternatives for a web content management system", Proc. of the 35th HICSS, 2002, pp. 2977—2986.

[12] C.M. Hurvich, and C.-L. Tsai, "A Corrected Akaike Information Criterion for Vector Autoregressive Model Selection", Journal of Time Series Analysis, vol. 14(3), 1993, pp. 271-279, https://doi.org/10.1111/j.1467-9892.1993.tb00144.x.

[13] M. Marques, J. Simmonds, P.O. Rossel, and M.C., Bastarrica, "Software product line evolution: A systematic literature review", Information and Software Technology, vol. 105, 2019, pp. 190–208, https://doi.org/10.1016/j.infsof.2018.08.014.

[14] A.M.J. Skulimowski, "A Foresight Support System to Manage Knowledge on Information Society Evolution". In: K. Aberer et al. (Eds.): SocInfo 2012. Lecture Notes in Computer Science, vol. 7710,. Springer, Heidelberg, 2012, pp. 246–259.

[15] A.M.J Skulimowski (ed., auth.), "Trends and Development Scenarios of Selected Information Society Technologies", Progress & Business Publishers, Kraków, 2018.

[16] A.M.J. Skulimowski, and I. Badecka, "Competition Boosts Innovativeness: the case of CMS evolution ", in: KICSS 2015, Advances in Intelligent Systems and Computing, vol. 685, Springer, 2018, pp. 188-204.

[17] A.M.J. Skulimowski, and I Badecka "Software Innovation Dynamics in CMSs and Its Impact on Enterprise Information Systems Development In: Tjoa, A.M. et al. (eds.) CONFENIS 2016, LNBIP, vol. 268,. Springer International Publishing AG, 2016, pp. 309-324.

[18] G.W Snedecor, and W.G Cochran, "Statistical Methods", Journal of Educational and Behavioral Statistics, vol. 19 (3), 1994, pp. 304-307.

[19] C. Young, and K. Holsteen, "Model Uncertainty and Robustness: A Computational Framework for Multimodel Analysis", Sociological Methods & Research, vol. 46(1), 2015, pp. 3-40.

[20] T. Yue, H. Zhang, S. Ali, and C. Liu, "A Practical Use Case Modeling Approach to Specify Crosscutting Concerns", in: G. Kapitsaki, E., Santana de Almeida (eds), Software Reuse: Bridging with Social-Awareness. ICSR 2016. Lecture Notes in Computer Science, vol. 9679, Springer, Cham, 2016, pp.89-105.