Data-Centric Decision Support

Rudolf Kulhavý
Honeywell Laboratories and
Institute of Information Theory and Automation
Pod vodárenskou věží 4, 182 08 Prague, Czech Republic
rudolf.kulhavy@honeywell.com

Abstract

The paper deals with both methodological and practical aspects of design, implementation and application of *data-centric* decision support systems powered by the historical process and business data. The paper is written from the product development and corporate R&D perspective and discusses major decisions and traps on the developers' way from the original idea to its commercial use.

1 Introduction

A *decision support system*, commonly abbreviated as DSS, represents a specific form of control system that suggests multiple possible actions (decisions) to the ultimate controller (decision-maker). The suggested actions are typically accompanied with quantification of their impact onto the controlled system.

The crucial feature of DSS is its capability to interact effectively with the user. DSS is considered mostly at a fairly high level of control hierarchy where the complexity of task or the importance of decision requires *human* intervention.

DSS often combines multiple functions, such as estimation, prediction, optimization, fault detection and diagnosis, and data visualization.

DSS incorporates both data and models. Given that large repositories of historical data have become nowadays a commodity by-product of computerized control, the interest naturally increases in *data-centric* concepts of DSS.

The paper summarizes the author's experience with design and implementation of a data-centric DSS in the process control context.

The challenges of DSS development are shown to go far beyond the underlying technology. The quality of data, a good dose of subject area expertise, and understanding of the total cost of ownership on the end user's side are crucial prerequisites for success.

The process of turning the idea into a commercial concept (product, service or technology platform) sets a separate challenge. The paper skips obvious though important aspects, such as effective project management and software development process, and focuses on the transfer of knowledge from the development team to a business unit.

2 Methodology

The selection of a proper approach to modeling, simulation and optimization is essential for a data-centric DSS, especially if it is to be applicable to a wider range of problems.

Modeling. The purpose of modeling is typically prediction or explanation. The following discussion assumes prediction to be a primary objective.

First-Principle vs. Empirical Models. In contrast to first-principle models, empirical models can explain how the controlled system behavior depends on the external causes or previous operation. Problems like product demand forecasting or condition based monitoring are hard to solve without fitting historical data using empirical methods. On the other hand, the empirical models cannot predict cases not met in the data history unless they are combined with other sources of knowledge, which is possible but at extra cost.

Linear vs. Non-linear Dependence. The behavior of real-life systems, especially at the high control level where DSS is applied, is rarely linear. To cope with global nonlinearity, the *linear regression* can be combined with a data discounting mechanism so as to fit the recent data behavior only. An option is to use a *nonlinear regression* model, such as a feed-forward neural network, to describe the global behavior of data. A compromising solution is represented by *non-parametric regression* that builds an individual regression model for each situation.

Adaptation vs. Learning. The classical regression with data discounting, which has become so popular in adaptive control and signal processing, does not actually learn from the data history; its adaptation is driven by the prediction error rather than a specific situation. If, e.g., multiple modes of operation need to be captured by the model, adaptive re-

gression can achieve that only at the cost of periodic retuning. This is in contrast to non-parametric regression that can build a model for purpose from relevant historical data. The nonlinear regression model—given the complexity of its estimation—can rarely be adapted in each step. In practice, it is reestimated from time to time while a fixed model is used in-between.

Complete vs. Recent vs. Relevant History. The global, local-in-time and local-in-space modeling paradigms—exemplified by the nonlinear, adaptive linear and non-parametric regression—work quite differently with the data history, making use of all, recent or just relevant data, respectively.

Closed-Form vs. Iterative Solution. It is of advantage if an analytic solution can be given to the data fitting problem as is in the case of linear-in-parameters regression, be it in global or local (non-parametric) setting. The estimation returns in this case a full description of the parameter and prediction uncertainty.

Recursive vs. Batch Processing. Much of the research initiated in 1960s and 1970s in optimal filtering and system identification was firmly locked in the computational paradigm of a recursive algorithm working over a low-dimensional data statistic compressing the previous history of data. This paradigm has lost nowadays most of its original rationale. The *just-in-time* computations of models built *for purpose* from data selected *on demand* has become a viable option.

Observed vs. Hidden Variables. A typical data-centric DSS uses methods that fit directly the observed data, such as different sorts of regression. There are multiple problems, however, where formulation using state-space models is more natural. Examples include processing of laboratory measurements or meteorological forecasts. Kalman filtering is a natural choice here, as a counterpart to global linear regression. Local estimation of multidimensional Markov random fields can become in the future an alternative to non-parametric regression.

Continuous vs. Categorical Variables. The support for hybrid models combining continuous and categorical (discrete) variables is relatively poor at the moment. A model combining the regression and Markov chain characteristics is needed. A specific difficulty in applying non-parametric statistical methods is definition of similarity between categories (different values of a discrete variable). In what sense is the operating mode "A" similar to the mode "C"? Or, Tuesday operation to Thursday operation?

Uncertainty vs. Reliability. A precise quantification of the prediction uncertainty is a key point for DSS to be perceived as a reliable advisory tool. Most users understand that statistical methods cannot predict reliably situations not met in the past, but expect at least a fair warning. In cases subject to high uncertainty—caused by the lack of historical data, too many variables entering the model, or too complex models—Bayesian model averaging represents a theoretically optimal solution, which mixes *all* individual predictors with *weights* given by the posterior probabilities of the respective models.

Prior Knowledge vs. Actual Data. In cases where the model-based prediction exhibits too high uncertainty, DSS can optionally combine the raw data with prior knowledge. Bayesian statistical methods allow for such knowledge straightforwardly, provided it is expressed in terms of a probability distribution of model *parameters*. This option is often cumbersome for non-statisticians. A more practical way is to ask about a typical distribution of the underlying *data*. Even then, the cost of eliciting prior knowledge is rather high and considered currently only for important and repeatedly applicable special cases.

Interpolation vs. Extrapolation. The previous point is related to the question how far from the past operation one can reliably extrapolate from the historical data. Does a data-centric DSS enable the user to "think out of the box?" The answer is given basically by the the choice of a model class. An extremely simple model (like fitting by constant) suggests local data interpolation whereas a more complex model (using higher-order polynomials or other suitable basis functions) can extrapolate far from the available data, provided we have other evidence (domain knowledge) that the model structure fits the problem.

Simulation. Black-box modeling gives typically poor performance for systems composed of multiple subsystems that can be combined in multiple ways. Think of a complex production schema composed of multiple processes with different dynamics. In order to capture the complete behavior of such a system, one would need to operate it in all possible configurations first. This implies a lengthy learning process—unrealistic in most cases. A faster and more direct way is to *model* the subsystems and then *simulate* their combined behavior.

Uncertainty Propagation. Simulation of systems composed of statistical models with uncertain responses require to generalize statistical modeling so as to make it capable to cope with uncertain values of independent variables. Markov Chain Monte Carlo methods offer general-purpose although computationally rather costly tools for uncertainty propagation.

Optimization. Optimization of systems with uncertain responses and for a wide range of objective functions requires a robust general-purpose method capable to handle the uncertainty. Stochastic optimization methods, such as simulated annealing, genetic algorithms or tabu search appear to be a good choice for static optimization. Approximate dynamic programming, reinforcement learning and Markov Chain Monte Carlo apply to the dynamic case.

Optimization vs. Enhancement. Decision tasks perceived as *hard* exhibit one or more of the following characteristics: lack of structure (resulting from combination of combinatorial, discrete, and symbolic variables), inherent presence of uncertainties (requiring time-consuming averaging), and huge search space (not easy to parameterize and prone to combinatorial explosion). In such problems, it is unlikely to find a global optimum in times available typically for decision analysis. Rather, the user expects to be navigated in the "right direction," resulting in enhancement of the current practice.

Uncertain Objective Functions. In decision support, one often faces the problem of optimizing performance indicators (such as the operating cost or profit) the actual values of which are known only *ex post*. This goes beyond the traditional formulation when the objective function is known and all uncertainty is due to the uncertain response of the underlying system. The expected rewards need to be predicted then before a decision can be made.

Multiple Objectives. The higher is the level of decision-making, the more often the objectives to be optimized are multiple and contradictory. Most theories of optimization with multiple objectives boil the problem down to optimization with a single objective function being a linear combination of individual objective functions. As the resulting objective function may sometimes be difficult to interpret, it is important to provide the user with a capability to analyze multiple (possibly all dominating) strategies.

Uncertainty and Risk. The uncertainty about the actual reward produces risk on the decision-maker's side. Consider two decisions that have the same expected reward but one is subject to higher uncertainty than the other. Which decision should be preferred? The question can be answered only after the decision-maker explicitly defines his or her attitude towards the risk. For one who is risk-prone, the high uncertainty promises additional gains. For one who is risk-averse, the high uncertainty threatens higher losses. The decision theory resolves the dilemma by asking the user to define an explicit utility function, which transforms the rewards by stressing either the gains or the losses.

3 Work Process

The actual data exploitation—using a combination of modeling, simulation and optimization methods—is only a fraction of the entire knowledge extraction process.

Data Preprocessing. Before the data can be exploited in DSS, it needs to be extracted from existing, often multiple data sources, integrated in one data repository, validated and cleansed by removing or correcting corrupt values, and eventually transformed (aggregated, filtered, scaled) as needed. This step takes 60–70% of time and resources of a typical DSS application project. Regardless of its importance, the task attracts a little attention of the research community, being perceived as mundane and routine.

Data Warehousing. The data-intensive methods, such as non-parametric regression, require data organized in a way that facilitates their quick retrieval. One option is to prebuild a database table so that it contains data for all the variables entering a specific statistical model. In principle, each new model is associated with a new database table. Such a solution results in high redundancy of stored data, but guarantees a quick access, especially if the table rows are properly indexed.

Model Building. The process of model development takes about 20–30% of time and resources and represents thus the second most demanding task. It starts by the selection of independent variables that significantly affect the system response or decision reward. Then the model structure is optimized. For non-parametric regression, the neighborhood shape (bandwidth parameters) is eventually tuned off-line or on-line. The model building process can be partially automated by running different models against historical data and comparing their performance. Complementing the automated search with a good dose of common sense (based on the domain knowledge) is always recommended.

Model Exploitation. It is here where all the power and beauty of statistical and optimization methods is applied. For a given model, their application is usually straightforward, assuming that enough attention was paid to the algorithms' numerics in the implementation phase.

Knowledge Presentation. The quality of graphical user interface and ease of use affects crucially the user's perception of the DSS usefulness. A specific challenge is how to present the model uncertainty and decision risk to a non-professional user without a statistical training.

User Guidance. DSS provides an effective support only if the user feels at every moment in control. In modeling, it is helpful if the user can check the results of data fitting graphically against the raw data. In optimization, the user should be given the possibility to suggest decisions around which DSS does a fine search.

4 User Acceptance.

The success or failure of DSS depends to a large extent on the user acceptance.

General-Purpose vs. Problem-Specific. DSS designed as a general-purpose tool makes the user save on the purchase price (the development cost is distributed over many copies) and multiple-product training (one tool can serve multiple tasks). DSS designed as a problem-specific application makes the user save on the model development (models are built in) and application training (single-purpose tool

speaking the domain language is easier to master).

Untrained User vs. Expert. *DSS for expert* is to *DSS for non-professional* as a racing car to a family van or as a surgeon's scalpel to a Swiss-army knife. It makes little sense to ask which one is better as they serve different purposes. It is not only the user interface, but primarily the choice of methods that makes the difference. DSS for the untrained user requires generic but robust methods that can be tuned quickly for 80/20 solution, with potential for further improvement at extra cost if required.

Process vs. Business Culture. DSS often requires data and knowledge from both process and business communities. The communities have different cultures; while the process people behave as asset owners, the business people act more like opportunity scouts. The different perspectives of these two classes of users can hardly disappear in DSS. Although the ideal solution remains to be one integrated system shared by both process and business people, such a system is still rarely met in practice.

Perception of Decision Support Itself. Much of the foundations of a data-centric DSS has been identified in the past few years with the area of *data mining*. A certain exaggerated hype, promising occasionally a new silver bullet, can hinder the acceptance of a data-centric DSS.

5 Product Development

When a decision is made what methodology will power DSS, how the data will be prepared and results presented, and who will be a typical user, time comes to implement the system.

Building a Team. There are significant differences between research and development as done in corporate R&D organizations and in technology start-ups. While the corporate R&D can generally benefit from bigger concentration of resources, it has a tendency towards distinguishing strictly between technology and product development, assuming a working *technology transfer* between both. Often, the assumption does not work; either the technology transfer has no obvious champion or the product development team has its own R&D capability and vision.

One option for teams in the corporate R&D is to emulate the multi-functional teams typical for technology start-ups, which combine development of new methods, rapid prototyping of software, pilot applications in various markets and technology marketing inside and outside the company. This way the technology transfer friction can be reduced while the development team preserves a strong feeling of ownership of the resulting product.

Software Implementation. The decision on the right way of packaging the technology is far from easy, in both user interfaces and software architecture. Should one focus a desktop tool or client-server architecture or Web deployment? Should one prefer performance or commonality? What set of software technologies to choose, given the uncertainty as to where the major vendors will be in a couple of years from now? And, most importantly, will the end user enjoy the result?

In the increasingly dynamic and uncertain environment, many development teams currently prefer a spiral (scrum) model of software development to the linear (waterfall) one. The scrum model assumes from the early phases involvement of a sample of target users and counts on evolutionary development with several iterations in software design and implementation. The objective is to capture the right product concept as quickly as possible, before the accumulated development cost prevents any further major changes to the software design.

Work Organization. Most of the risks down the road are about the people and due to the people. The whole development goes typically in a highly competitive environment. There is no control strategy that would guarantee automatic success. Nevertheless, there are some "good practices" worth following.

Communicate vigorously. If you are to choose between one more software feature and one more person who likes your work, go always for the latter.

Schedule your development carefully. Do the right things at the right time. Be prepared that at the early phases it is often an intuitive rather than a fully controlled process.

Allocate your resources wisely. There are always more things to do than resources available. Find a balance between the product maturation and market growth.

Prototype software quickly. If you are not 100% sure that you develop the right thing, do it fast at least.

Approach development as evolution. Most users prefer functionality to performance if they are forced to choose one. Start with the former, you may be given time to optimize later.

Productize successful solutions. Make running solutions configurable and reusable rather than the other way around. Resist the temptation to proceed in a linear way.

6 Growing a Business

To see that a highly educated team makes an innovative solution work is exciting in its own right, but it does not necessarily sets a pattern for a successful business. A natural concern of a business unit is whether its application engineers or a typical end user will be capable to solve similar

problems on their own, without direct involvement of the developers.

An answer to this question depends on the level of complexity and sophistication of the problem to be solved, the software tool available and the problem solver and tool user in one person. There are several basic ways how the end user can be supported in solving a nontrivial problem using DSS tools.

Workflow Automation. Knowledge-intensive steps in DSS configuration can be automated to some extent. A typical example is partial automation of the model selection process based on a built-in capability to assess a particular set of models against a particular set of data for a particular set of situations. Needless to say, this extra capability does not come for free; it requires more development effort, more computational time and a good dose of experience (or training) in interpreting the results.

Solution Library. Experience from multiple projects can be effectively shared through a library of proven "template" solutions, which can be reused with relatively little effort. The library lists model structures recommended for frequently met problems and indicates their typical performance. A solution library provides an effective way of sharing knowledge between application groups and the development team.

Modeling Services. The supplier of DSS system can consult the customer on an appropriate model for a specific problem. This is typical when delivering a complete solution, which will be further maintained by the supplier. The model may still require periodic retuning, performed by the software itself or as a remote service.

User Training. Consulting on model development assumes that the user comes back whenever he or she faces a new problem. Some customers may find more cost-efficient to expose their frequent users to advanced training. The case when the end users are supposed to develop models on their own represents a major challenge for developers who must position themselves on the scale between simple widespread tools such as spreadsheet calculators and specialized statistical and optimization packages for expert users. It goes back to the decision about the target end user.

7 Innovation Story

The beauty of hindsight is that it makes it possible to organize the development steps in an linear and logical way. In reality, the innovation and development process goes rarely a straight way.

Initial Idea. For a team in Honeywell Laboratories, the DSS endeavor started when confronted with the task to forecast total steam, heat and electricity demand, heat demand in individual hot-water pipelines, and total gas consumption in a municipal district heating system. The system was relatively complex, composed of five generation plants, a steam pipeline network totaling 60 miles, and five primary, partially interconnected hot-water pipeline networks of total length of 46 miles. Information about the system was too crude to build a simulation model. The only information available to the team was a set of historical production and meteorological data for about one year. Under these circumstances, the team was naturally tempted to describe the system behavior through an empirical model.

The attempts to apply classical regression with age discounting of past data failed because of a strongly nonlinear behavior and a large portion (15–20%) of missing and unreliable data in the archive. A global nonlinear model such as a neural network was abandoned soon, too. The demand profile was found to change rather quickly in the transient economic conditions of the country. As a result, the global model needed to be periodically retrained, which significantly compromised the forecast quality towards the end of period. In addition, the periodic adaptation of model meant to develop an extra application running in background in order to meet the customer's requirement of 24x7 operation of decision support.

Since none of the traditional approaches worked satisfactorily, the team decided eventually for a hybrid approach—applying a multiple regression model to only a fraction of the past data points that were "similar" to the forecasted situation. This worked remarkably well and became a basis for a software prototype that was installed and commissioned successfully at the customer site.

Technology Maturation. Soon after a working solution to the problem was found through experimentation, a number of similarities were recognized with known methods, such as locally weighted regression in non-parametric statistics [1], memory-based or lazy learning in machine learning [2] and just-in-time or on-demand modeling in system identification [3].

These connections were used to mature quickly the prototype solution and to lay down a solid basis for its further elaboration. Local regression was complemented with local classification for the case when the target variable was categorical. Models mixing continuous and categorical independent variables were supported. The frequentist formulation of the estimation problem, favored traditionally in non-parametric statistics, was replaced with a Bayesian framework [4] allowing for incorporation of simulation- and expert-based knowledge [5]. A lot of research effort has been put into providing a reliable capability to do model assessment and selection off-line and on-line.

A significant driver for innovation became the fact that the local modeling concept was applied to a huge operational database storing low-quality data. Apart from the need for

continuous data validation, the team started identifying and transferring relevant knowledge from the data warehousing community. A theoretical support for linking data warehousing and Bayesian data smoothing was provided by the information geometry of parameter estimation [6], originally developed in a global modeling context.

Scope Stretching. With local modeling in place, the team was in position to develop "localized" versions of optimization and diagnostic tools working only with a fraction of relevant historical data points. The work focused in particular on local algorithms for process enhancement and abnormal event detection. In addition, state-of-the-art tools for visualization of multidimensional data were implemented and made part of the package to support further the model building process.

All the tools continued to share a common paradigm:
1. Retrieve historical data relevant to the case.
2. Fit the data with a model of appropriate structure.
3. Use the model for forecasting or decision-making.

A common paradigm made it possible to preserve a small kernel of general-purpose functions that could be reused effectively for multiple purposes.

Software Implementation. It took a few months in a couple of people to develop the first instance of a data-centric forecasting solution. It took several years in a considerably larger team to turn it into a fully configurable product prototype.

Early decision support or data mining products were designed for highly educated and experienced users. The team's ambition has been to come up with a tool that could be used by non-experts, with little or no prior training. The search for the most appropriate way of packaging the underlying technology resulted in several internal releases of software that applied different architectures and user interfaces.

To speed up the development in the early development phases, the team used several rapid prototyping platforms. Later, when the integration with the business units' products became a priority, the team standardized in software development tools with the divisional teams.

A challenge of its own was to develop quickly data warehousing skills and adapt the technology to warehousing of process data.

Knowledge Transfer. The team focuses currently on making the technology and software usable by non-experts and on expanding the scope of applications. Discussion goes increasingly about the business aspects of further development and effective ways of sharing the knowledge.

The development would not be possible without a strong and cohesive team with a start-up mind-set and firm conviction in the underlying idea. In building the team, it was decided to prefer generally highly educated and eagerly learning people to highly productive but narrowly oriented specialists. Looking back, it has turned out to be a lucky choice. In the team where more than 80% people carry a Ph.D. degree, most have changed their job descriptions several times during the development and become active contributors to further development of the technology, software and industry-specific solutions.

8 Conclusion

The paper presents experiences from development of a data-centric Decision Support System that takes advantage of synergy of data warehousing, non-parametric statistics and stochastic optimization technologies. The technology issues are shown to be a crucial but—measured by the overall productization and commercialization effort—a relatively small part of the whole endeavor. A successful control of the innovation process requires to view the technical part of development from the overall system perspective, which takes into account the conflicting interests of all major stakeholders and ultimately benefits the end user.

References

[1] W.S. Cleveland, "Robust Locally Weighted Regression and Smoothing Scatterplots," *J. Amer. Statist. Assoc.*, vol. 74, pp. 829–836, 1979.

[2] L. Bottou and V. Vapnik, "Local learning algorithms," *Neural Computation*, vol. 4, pp. 888–900, 1992.

[3] A. Stenman, "Model on Demand: Algorithms, Analysis and Applications," Ph.D. Thesis, Dept. of EE, Linköping University, No. 571, 1999.

[4] V. Peterka, Bayesian approach to system identification, in P. Eykhoff (ed.), *Trends and Progress in System Identification*. Elsmford, NY: Pergamon Press, 1981, pp. 239-304.

[5] R. Kulhavý and P. Ivanova, Memory-based prediction in control and optimisation, in *Proc. 14th World Congress of IFAC*, Beijing, PRC, vol. H, 1999, pp. 289-294.

[6] R. Kulhavý, *Recursive Nonlinear Estimation: A Geometric Approach*, London: Springer-Verlag, 1996.

Acknowledgments

The author would like to acknowledge the work of the Data-Centric Technology Team of Honeywell Laboratories. The work presented in the paper would not be possible without an effective teamwork.

The author's research has been supported in part by the Grant Agency of the Czech Republic through Grant 102/01/0021. The support is gratefully acknowledged.