Continuous Execution of System Dynamics Models on Input Data Stream

Ivan Perl
ITMO University
Saint-Petersburg, Russia
ivan.perl@corp.ifmo.ru

Alexey Mulyukin, Tatyana Kossovich
ITMO University
Saint-Petersburg, Russia
alexprey@ya.ru, tatyana_kossovic@mail.ru

***Abstract*— This article describes a new approach for system dynamics models execution. In most cases when model execution is involved it is performed on a set of static and known data, which are sent to the model as an input. And it is expected, that on the model output modeler will get a set of other system or event characteristics, computed by the model based on the input parameters. This approach still has the widest usage, but it is not the only one scenario, which is demanded by different industries. With growing popularity of concepts such as Internet of Things, demand in modeling based solutions, which take as input continuous data streams, has grown significantly. In comparison with stand-alone client-side modeling systems, cloud-based solutions, such as sdCloud, became a reasonable answer to such industry request. Such systems can provide an ability of continuous execution of system dynamics models. In other words, these systems are ready to accept an incoming data stream and perform model execution that will result in streaming modeling results back to the end-user. Running system dynamics models in parallel with the process it is describing allows to perform predictive modeling of the system status in the future, and it also allows to find additional hidden external impacts to the model. For example, such approach can be a base for predictive maintenance of complicated technical systems, because it allows computing nearest maintenance time more efficient.**

I. Introduction

Business Dictionary provides following definition of system dynamics: “General purpose framework for depicting and simulating fluid (dynamic) behavior of business, economic, environmental, and social mechanisms”. Nowadays, this approach obtains wider applications to describe also complicated technical systems like IoT and other Big Data oriented areas. For many applications, it is not enough to be able to run the model on a set of already known parameters to define values of another set of characteristics of a system or an event.

Talking about models execution in general and system dynamics models in particular, we assume that we have formatted model definition itself and execution engine, which can run this model. As a start point for modeling process, we take input data stream. In most cases, such modeling input data represents several known properties of the system, and so, basing on them modeling is performed to identify set of undefined system properties. During development or debugging of a model the most common situation is when historical data is taken both as model input and model output. Since all the properties are known, modeler can judge how well his model operates, and then he can do the necessary tuning until difference between expected results and model execution results becomes close enough for the interested problem area.

Such approach works perfectly when we have enough historical data describing the system and our goal is to be able to predict its behavior in the future or perform a deeper investigation of its internal processes. Frequently modelers use local modeling environment for such operations. In other circumstances, like modeling of complicated systems or in big companies, they use shared modeling environment on powerful server. Adjusted for the current trend of moving different solutions into the cloud, different kind of modeling is affected as well. Along with paradigms like Platform as a Service (PaaS), Database as a Service (DBaaS) and Software as a Service (SaaS), the new entity called Modeling as a Service (MaaS) appeared. In general, model execution is set to execute on the local environment, but cloud-based implementation could bring many new features, which could improve basic modeling concepts and bring cloud-based modeling to the new level [1].

II. Motivation

Day by day, the complicity of our world is rapidly growing, and it leads to the growth of model's complexity because it has to describe a lot of different aspects with a high enough precision to achieve a reasonable and correct decision. Along with the complexity, an ability to gather a different kind of data is growing as well, and this trend leads us to deal with big amount of collected data, which should be processed. The special scientific area called Big Data has become the most popular for collecting, analyzing, storing and processing for all kinds of huge amounts of data. For sure, such trends cannot put modeling industry aside and cloud-based MaaS solutions has become a reasonable answer to the modeling challenges in terms of Big Data world.

By its nature, Big Data is not only related to a large amount of data stored in data centers, but it also related to the intensive data streams that are passing between different systems. This important aspect should be considered while moving modeling solution to the cloud and converting them to MaaS solutions. As an example, we can take an Internet of Things (IoT) solution. In [2] it is shown, that those system dynamics methods are well suitable in different areas where system IoT is applied. One of such aspects is IoT-enabled public transportation: “System Dynamics is well suited to identify the behavior of particular systems and also provides a framework for theoretical analyses for researchers to investigate the sensitivity of systems to structural changes. Several authors have used System Dynamics to investigate various facets of transportation systems [3], [4], [5], [6]".

IoT solutions involve millions of connected devices and are able to produce extremely high volumes of data that should be persisted and analyzed. While dealing with such data stream their analysis became a critical feature for the whole solution. It is useless to gather terabytes of data from any deployment in several minutes, because of people are not able to process so much data. However, it might be a smart intelligent system that is able to aggregate and understand data collected from all the devices and draw users' attention only to important events and changes. Thus the usefulness of such system will increase significantly.

To solve many IoT-related problems it is required to have a special mechanism allowing prediction of the direction in which system state is changing. In [7] application of system dynamics modeling for IoT area described in following way: "The purpose of System dynamics modeling is generally classified into two ways. The first purpose of modeling is for understanding or describing the past behavior of the system. To understand the past behavior of the system, modelers describe Behavior over Time Graph. On the other side, system dynamics modeler simulates the model to prescribe or forecast the future based on the current structure of the system. These two approaches of modeling are often conducted simultaneously." For example, IoT-enabled traffic monitoring system can easily answer the questions like "What traffic we had on last Wednesday?" or "What is the current traffic situation in district ABC?". Answers to these questions are already present in the giant data volumes that are gathered from cars onboard sensors and smart sensors on the streets. More interesting is to find an answer to the questions like "What will happen if road XYZ will be closed for roadworks on Wednesday from 11 am to 4 pm?". And in this case, depending on the complexity and specification of prediction mechanisms, term "happened" can be treated widely and include aspects like:

- Expected traffic speed on adjacent streets
- Expected delays for public transportation in current or surrounding districts
- Expected impact for emergency services
- Expected changes in air pollution because of possible traffic-jams

III. Predictive Maintenance

However, this list contains just a high-level, let us say, generic impacts. There can be additional area-specific answers to the question above, that make sense for the concrete area. For example, an owner of the traffic company, who has an IoT related bus fleet or another kind of public transport. And he must take care of his assets in a proper way because of maintenance that should be performed properly and on time. Otherwise, it would take much more money to replace broken of damaged assets with new ones. One of the most popular IoT applications is called Predictive Maintenance (PM). The PM concept means that we gather information about the usage of our asset and try to perform its maintenance in a most reasonable timeframe when it is required. We don't rely on a simple schedule "Maintenance every 30 days", but we rely on more precise things like "Maintenance every 1000 engine running hours" or the more complicated combination of parameters that can be monitored. If we will proceed with the analysis of the example with planned road works, our goal is to be able to predict how this situation will affect maintenance plans. Most of the times there are two answers to the roadworks challenge on the street: to keep the same route (if it is possible) and agree with lower traffic speed or to find a detour. Assuming that initial route was optimal, any kind of detour will lead to the longer route resulting in more working hours for engine and more distance to be run for the car chassis. This can be represented by the following schema.

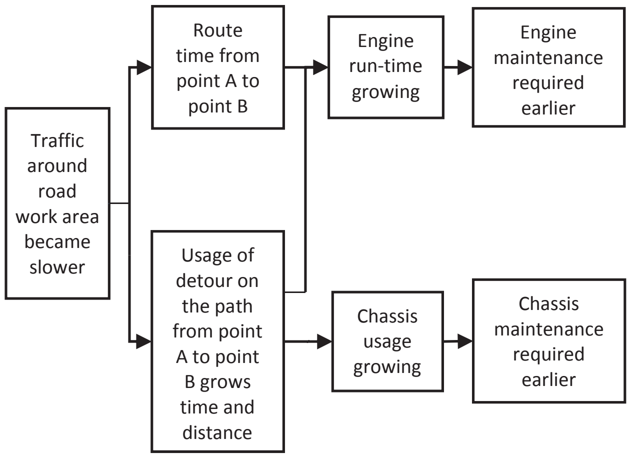
Traffic around road work area became slower
Route time from point A to point B
Usage of detour on the path from point A to point B grows time and distance
Engine run-time growing
Engine maintenance required earlier
Chassis usage growing
Chassis maintenance required earlier

Fig. 1. Options to handle road works and their effects

IV. Understanding of continuous modeling

One of the ways to do the prediction of the system behavior is to collect historical data and perform modeling of the required period of time. This kind of prediction often works when we deal with systems that repeat their behavior with some linear or not too complicated non-linear coefficients. In other words, we may say that traffic in some district, let us say, between 7 am and 10 am is almost the same every day because people mostly go to work and this repeats regularly. To improve the precision of such model we may split week into workdays and weekends because patterns in these kinds of days differ. Diving deeper we may differentiate workweeks and say that all Mondays are alike, on Tuesdays are alike and so on. This will allow considering regular events like fuels delivery to the gas station on Tuesday and Friday at 8:30 pm, which leads to the small traffic complications on some cross road caused by proper parking of the truck at the gas station. Next step could be splitting days by actual date. Such separation will allow considering events, which take place, let's say each 11-th and 25-th day of the month. But, this specification significantly reduces statistics base that can be taken. When we assume that all days are same – we have 365 reference points during the whole year. If we assume that only workdays have the same behavior – we have only 250 references to one year. But if we assume data group by day of week we will have 52 references for each day of week and in case when we group statistic data by number of days in month, it will be only 12 days to one year. On the other hand, if we will try to extend data collection window from one year to, that is to say, three years, we will include so many hidden parameters affecting statistics that it will be hard to work with such trends.

Besides statistical methods, there are other approaches to make reasonable predictions. Most of the IoT systems are dynamic systems evolving over time. That is why this fits borders of system dynamics very well. There are many works related to the development of system dynamics models to cover various aspects of different IoT-enabled systems. From the system dynamics perspective, it is very important that IoT comes to many areas that knew and investigated for a long time. For example, traffic planning and monitoring. Being a system dynamics issue, traffic planning studied in many different works and described with well-designed and examined models. IoT has settled down in this area recently and it is relatively easy to benefit from the synergy of the existed studies and ability to gather big amounts of actual data.

Initially, system dynamics models executed on data sets that were somehow historically gathered. For IoT-enabled systems, such approach is still easily applicable. For instance, we can gather data for 24 hours and then run a model with this baseline to predict next 24 hours. Since IoT data is an endless stream, it will be much more profitable to introduce new paradigm for system dynamics models execution – continuous model execution on a real-time data stream. This paradigm matches MaaS approach especially well when model is continuously running in the cloud providing actual real-time forecasts depending on the incoming data stream.

Diagram on figure 2 shows a flow chart of the system dynamics model execution on a real-time data stream.

Such continuous modeling approach provides not only continuously updated forecast for the system, but also has two control loops for model's incremental tuning and behavior verifying. The first control loop is designed for continuous comparison of input data approximation with real IoT data which come to the modeling cloud with a delay. The second control loop is designed to do the comparison of three resulting streams at a time. For example, we are going to approximate input data for the next 24 hours and run system dynamics model with this approximated data in order to make a prediction about future state of the system. After 24 hours we will perform model computation on a real data delivered from the system, and after all, we have an opportunity to compare predicted result to real one. This comparison will show how close modeling results (both computed on real data and predicted) matches to the real system state. Also, it is possible to do a real measurement from time to time and compare those actual results with modeling on real data and with the forecast. This comparison will show how close modeling results (both computed on real data and predicted) appear to the real system state. But, prediction of system state and behavior in the future is not the only application of the described modeling approach. Model, which runs in parallel with the real system could be treated as a flexible shadow system and allow looking for the questions, like "What if …?", etc. On the one hand, at any point in time, model keeps its last state computed on real data which is defined by the system's organization, along with operational system and model operational history. And from that state from the already described case, the model was computed to the future and this computation is based on the approximation of historical model input data. As an addition to this forecasting simulation, it is possible to feed model with any artificial input data stream that could be a simulation of some new usage scenario for the target system. In this case, the model will give an answer how the system will behave if (taking into account its current state) we will modify its usage scenario. This use-case will be valuable for planning various changes to the real system by playing with its shadow.

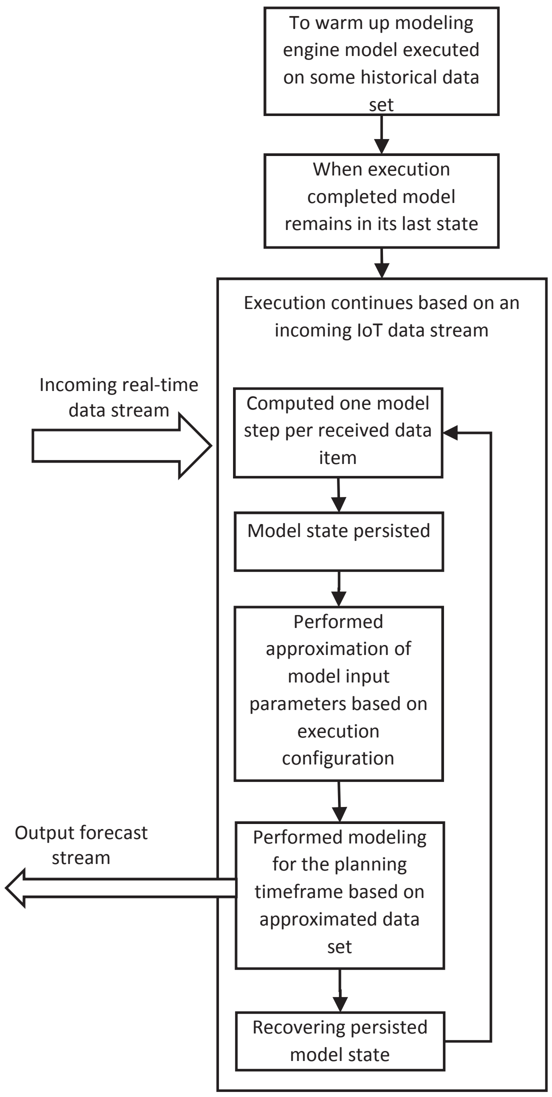
To warm up modeling engine model executed on some historical data set
When execution completed model remains in its last state
Execution continues based on an incoming IoT data stream
Incoming real-time data stream
Computed one model step per received data item
Model state persisted
Performed approximation of model input parameters based on execution configuration
Output forecast stream
Performed modeling for the planning timeframe based on approximated data set
Recovering persisted model state

Fig. 2. Basic flow chart for system dynamics model execution on real time data with input data approximation for predictive modeling

One of the areas of IoT use cases family, where such modeling approach is valuable is predictive and preventive maintenance. Such proactive approach for management of complicated systems is widely described in the literature

[8], [9]. According to [9], predictive maintenance is described in a following way: “Predictive maintenance is the complement of preventive maintenance. Through the utilization of various nondestructive testing and measuring techniques, predictive maintenance determines equipment status before a breakdown occurs. With predictive devices currently available, it is incumbent upon maintenance organizations to include the process of predictive maintenance in their maintenance programs. A total PPM program is absolutely essential to an efficient, reliable and safe production process. Benefits are direct and substantial, including: high product quality, long machine life, avoidance of work stoppage, high safety, high morale and fewer frustrations.”

On the other hand, the area of predictive maintenance was designed to be well described by methods and technics of system dynamics. Works like [10], [11] describe the application of system dynamics models and approaches for efficient planning and performing predictive maintenance. Here is how the application of this approach is described in [13]: “System Dynamics is a method which has established to analyze and model complex dynamic systems. The developed models do not only enable the user to compare the efficiency of different maintenance activities with each other, but also provide the possibility to determine a combination of these activities leading to an overall system maintenance strategy at a cost minimum through optimization. This allows infrastructure managers to evaluate their current maintenance program and to determine an economically improved strategy for the future.”

Let’s assume that system we simulate has two groups of parameters: input modeling parameters and output modeling parameters. Both are measured and taken from the real system at some moment. When we use continuous modeling approach, we measure input modeling parameters and stream them to the modeling cloud service. In the cloud service for each new set of data next model state computed. After that, modeling system performs extrapolation of the input modeling parameters for the period, which is used for further forecasting by computation and modeling performed per generated data set. As a result, we have output model parameters based on the extrapolated input stream. In other words, we have three data streams:

- Input modeling parameters which come from the real environment
- Extrapolated input modeling parameters
- Output modeling results

The first control point could be described in following way. Let's say we do a forecast for the traffic system for next 24 hours. It means that if we have done an extrapolation at moment T1, we would be able to compare extrapolated data with real one after 24 hours. This comparison will provide us an information about the quality of our extrapolation approach of how close generated data matches to the real data gathered during next 24 hours.

The second control point is the comparison of measured system parameters to the forecast provided by the system. Per result of model execution, we are able to get a vector R containing the system properties, representing some of the system parameters in the next 24 hours. After 24 hours, we can measure actual system parameters R’ and we will have vector R" which is the result of model computation on real IoT data. The difference between these three vectors will tell us how good our model is. Analysis of this delta may help to understand how model could be improved.

There are two options, how real-time data can be attached to the model: in synchronous way and in asynchronous way.

Synchronous way means that full set of input modeling parameters is delivered to the model in a single data set. In other words, if we have N parameters required to compute next model step, it means that all N parameters delivered to the model execution engine in one message where all required properties defined.

Asynchronous way means that different modeling properties streamed towards to model execution engine separately from the real system. And it is required to build a single data set for the next model execution step on a model execution side to align asynchronous changes in arriving data. Usage of different strategies for input stream synchronization can solve this problem. One of the simplest approaches is to keep most recent values on model inputs and use separated channel as a synchronization impulse for starting the model execution. This approach could safe more traffic channel between sensors and execution platform, because we can send only changed values.

Taking into account common approaches for application of system dynamics modeling in a field of predictive maintenance, we will compose a simple model describing dependency oil quality in car engine, based on car usage metrics in order to illustrate continuous modeling approach.

Such model could be widely used by different kinds of companies for efficient planning of oil change maintenance for their fleet items. Oil change date depends on several car usage parameters, such as number of engine running-hours, average speed, and carload. In most cases when predictive maintenance is not applied for fleet monitoring and management, fleet maintenance, including engine oil refresh performed on a calendar basis. It means that for each vehicle there is a set of fixed maintenance dates, which are not correlated with actual fleet item usage. Per statistics more than 25% car engine failures caused by low quality of engine oil. But when fleet item usage is not considered there are two critical options appear: oil would be changed when it could be used still, so it is inefficient, or it would be changed too late, so then it will have negative impact to engine itself.

Predictive maintenance based on a system dynamics model helps to forecast quality of engine oil based on actual usage information of a fleet member. Since major role of oil in an engine is to protect it pieces, heavier usage of the engine will result in quicker wear of oil. This concept was put into the illustrative model used in the article. Following schema depicts system dynamics model for described use-case.

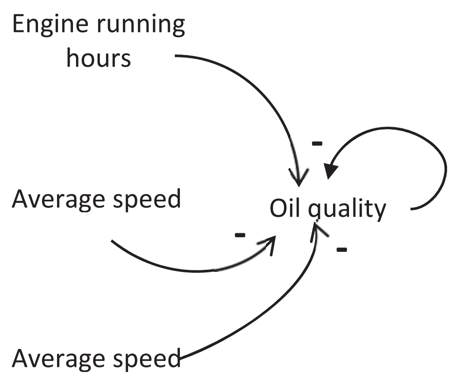

Fig. 3. Simplified system dynamics model showing dependency between car engine oil quality and car usage metrics: engine running hours, average load and average speed

All three properties have a negative impact on oil quality. Also, computed oil quality also affects itself, because in real system engine oil degradation is non-linear process and so as low oil quality is, as quicker it became worth.

To perform a validation of described approach for running system dynamics model on a real-time data for prediction purposes, we will use model for prediction of engine oil quality, described above. For our evaluation, will take the following scenario. An experimental car is equipped with two sensors: one reports average speed during last minute and another reports current carload. The car is driven according to its route of delivery different kinds of goods; this explains why its load level changes over time. Data is gathered every minute and delivered to the server every hour. That is why during modeling process we have two-time frames, during first, model is running on date that was delivered from the car, it is set to real IoT data. During next hour there is no data delivered from the car, so model runs on data set built as an approximation of historical data (approximation based on previous hour data), so modeling process turns into forecasting. In this experiment, we see two key points, in which we will validate modeling and approximation results. Next two charts show difference between real speed, carload, and computed approximations:

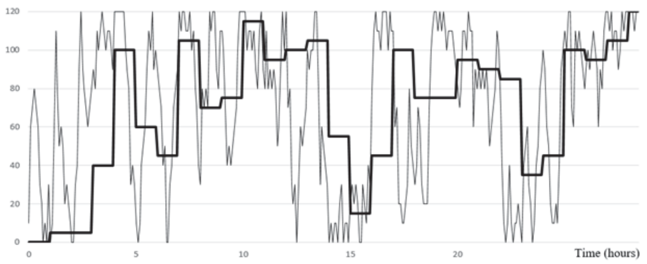

Fig. 4. Car speed (real and approximation)

Here, lighter lines show real IoT data that was gathered and darker ones is approximation used for modeling for periods, where IoT data were not available yet.

On the next chart, we can see the difference between modeling results for engine oil quality based on real IoT data versus modeling results that were based on approximation of input data for the potential prediction of the target value.

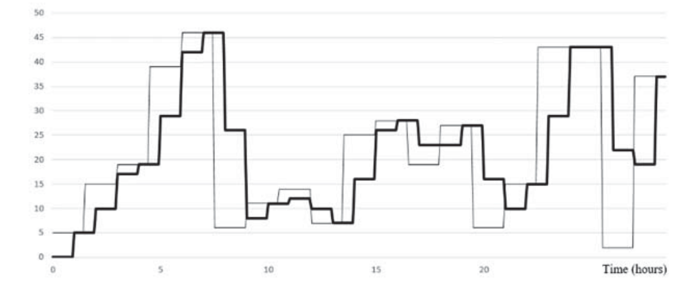

Fig. 5. Car load (real and approximation)

Here again, in lighter color we have modeling result on real data and darker one is prediction based on data approximation.

For this particular example, trivial approximation function was used. Approximation based on simple average of two values taken in the beginning and the end of the previous period. Even with such simple approximation approach, we can see that modeling result on real data and approximation are very close to each other.

Described use-case shows simplified approach for running system dynamics models on real time data, mostly, because of very simple approximation approach and very short period taken for running model as a prediction mechanism. In case of real usage of proposed approach additional research should be done for proper identification of approximation function and system dynamics model tuning based on analysis of deltas between prediction provided by model on approximated input data and model result computed on a real data after a while when predicted moment really comes.

Here is a basic workflow required to run a system dynamics model on a real data stream in cloud solution sdCloud.

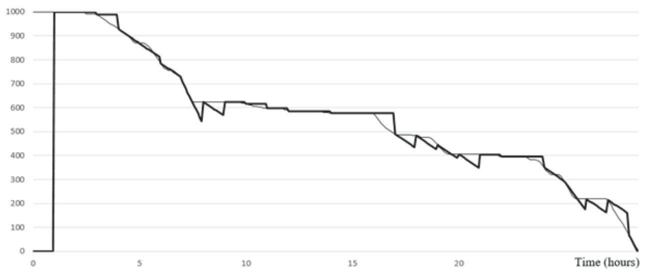

Fig. 6. Predicted oil quality (on real and on approximated data)

Described modeling approach for continuous execution of system dynamics models on a data stream provides following list benefits:

1) *Out of the box cloud and big data oriented modeling approach.*

2) *Ability of continuous monitoring of difference between real data stream values and approximated values, generated for predictive modeling.* If model execution is well tuned in

the beginning, for most technical systems behavior could be approximated with relatively high accuracy. If it will be detected, that over time delta between approximated and real values is changing significantly, it means that something unexpected or unpredicted happening with the system. Therefore, this metric could be a good sign to show that this system needs operators attention. Fox example, in that case with oil quality in the car engine, described above, changing of the delta in average speed between real reported speed and its approximation means that car cannot drive with the speed it was able before. For example, because of traffic changes due to roadworks or due to some issues with the car itself. In both cases, attention is required to return system to its previous working state.

3) Ability of continuous monitoring of difference between actual system measurement of system parameters and model execution results for real data streamed to the system, predictable modeling based on approximated data stream for the same point of time. If we see that modeling results have started to differ from real measured system parameters, it means that due to some reason model doesn't reflect system behavior anymore and it is important to figure out the root cause of such situation. For example, in the case described in the article, real measurements may show the lower quality of engine oil than forecasting calculations predicted or execution on real data shown. It could mean that new factor appears in the system and impacts on oil quality. For example, there is some engine issue, like dirty oil filter.

V. CONCLUSION

Modeling of dynamic systems is a powerful tool for their monitoring, prediction, and planning. Nowadays, many systems became internet-connected, for example, in IoT paradigm. When a huge number of devices, most of which were well known and were working offline became connected to the Internet. From one side, such shift leads to an ability to gather more information about actual systems behavior what can leads to better managed, optimized and efficient systems. But on the other side, this also provides high volumes of data which became as hard to manage as big they became. To get more value from the collected data it is important to convert raw data into the information, that will help to make proper decisions. When we are talking about data, we may say that yesterday temperature outside was T1 and tomorrow it is T2, but while it is raw data it is hard to make any significant decision.

For a long time, different kinds of simulation and mathematical models were used to understand how complicated systems are behaving and what are the ways to improve them. With continuous system dynamics modeling approach described in this article and implemented by sdCloud platform new application for them are revealed. Instead of running the model with some theoretical inputs and finding answer to the questions like “What if we will have this input?” or “What kind of input will lead to the required goal?”, continuous modeling helping to find an answer to the question “Where this particular system is going right now?”. In other words, continuous modeling is an approach for understanding data and translating into the valuable information, required for proper and efficient decision making.

Another benefit of such approach is that such kind of model is still obedient instrument allowing to verify “What if?” hypothesis. Since model is running on real data coming from actual system it became its shadow copy helping to figure out, for example, what will happen with our system if we will change this parameter or that and the answer will be related not so some system described by the model, but to the real system user dealing with.

This application considered by the sdCloud solution to turn almost sixty years of system dynamics evolution in the direction of modern cloud technologies and Big Data systems. Continuous modeling provides not only an ability to make a model of some system and compute unknown aspects based on known but also perform different kinds of system evolution monitoring and control over time.

REFERENCES

[1] A. Mulyukin, T. Kossovich and I. Perl, *Effective Execution of Systems Dynamics Models*, in 19th Conference of Open Innovations Association FRUCT, University of Jyvaskyla, 2016, pp. 358-364.

[2] Marshall P., *System dynamics modeling of the impact of Internet-of-Things on intelligent urban transportation*, Regional Conference of the International Telecommunications Society (ITS), Los Angeles, CA, 2015

[3] Abbas, K. A. and Bell, M. G. H. *System Dynamics Applicability to Transportation Modeling*, System Dynamics Conference, Transportation Research Part A: Policy and Practice, Volume 28, Issue 5, pages 373-390, September 1994.

[4] Bernardino, J. P. R and van der Hoord, M. *Parking Policy and Urban Mobility Level of Service – System Dynamics as a Modeling Tool for Decision Making*, European Journal of Transport and Infrastructure Research, Issue 13(3), pages 239-248, September 2013.

[5] Mitrea O. *System Dynamics Modeling of Intelligent Transportation Systems Human and Social Requirements for the Construction of Dynamic Hypotheses*, Selected Topics in Nonlinear Dynamics and Theoretical Electrical Engineering, pages 21-226, Springer, 2013.

[6] Hennessy G. et.al.: *Economic Dynamics for Smarter Cities* Proceedings of the System Dynamics Society, 2011.

[7] Eun K. L. Dong-Hwan K.,Won G. H., *System Dynamics Modeling for the Future IT Development: Applied to Education, Health Care, and Smart Work System in Korea*, Proceedings of the 29th International Conference of the System Dynamics Society, 2011.

[8] *Preventive and Predictive Maintenance*, 700ZB00102, https://www.lce.com/pdfs/The-PMPdM-Program-124.pdf

[9] H. M. Hashemian, Wendell C. Bean, *State-of-the-Art Predictive Maintenance Techniques*, IEEE TRANSACTIONS ON INSTRUMENTATION AND MEASUREMENT, VOL. 60, NO. 10, 2011

[10] Adolfo C. M., Ruiz U., *Learning Maintenance Management through System Dynamics*, Proceedings of the 13th International Conference of the System Dynamics Society, 1994

[11] Böhm, T. et al. *Efficient Maintenance Strategy Through System Dynamics* Computers in Railways XI (2008): Web. 7 Feb. 2017.

[12] J. Sterman, *Business dynamics: systems thinking and modeling for a complex world*, Cambridge: 2002.

[13] J. W. Forrester, *The Beginning of System Dynamics*, Cambridge: MIT, 1989.