

Helping Teams to Help Themselves: An Industrial Case Study on Interdependencies During Sprints

Jil Klünder, Fabian Kortum, Thorsten Ziehm, Kurt Schneider

▶ **To cite this version:**

Jil Klünder, Fabian Kortum, Thorsten Ziehm, Kurt Schneider. Helping Teams to Help Themselves: An Industrial Case Study on Interdependencies During Sprints. 7th International Conference on Human-Centred Software Engineering (HCSE), Sep 2018, Sophia Antipolis, France. pp.31-50, 10.1007/978-3-030-05909-5_3 . hal-02270699

HAL Id: hal-02270699

https://inria.hal.science/hal-02270699

Submitted on 26 Aug 2019

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.



Distributed under a Creative Commons Attribution 4.0 International License

Helping Teams to Help Themselves: An Industrial Case Study on Interdependencies During Sprints

Jil Klünder[1][0000−0001−7674−2930], Fabian Kortum[1], Thorsten Ziehm[2], and Kurt Schneider[1]

[1] Leibniz University Hannover, Software Engineering Group
Hannover, Germany
{jil.kluender, fabian.kortum, kurt.schneider}@inf.uni-hannover.de
[2] Arvato SCM Solutions
Hannover, Germany
thorsten.ziehm@arvato.com

Abstract. Software process improvement is a very important topic. Almost all companies and organizations face the necessity for improvement sooner or later. Sometimes, there is obvious potential for improvement (e.g., if the number of developers does not fit the project size). Nonetheless, fixing all obvious issues does not necessarily lead to a "perfect" project. There are a lot of interdependencies between project parameters that are difficult to detect – sometimes due to the influences of social aspects which can be hardly grasped.
We want to support the process of improving daily work by simulating and visualizing how project parameters evolve over time. Our approach is based on building a System Dynamics model that takes into account key performance indicators as well as assumptions about social aspects. In the present case, we chose parameters of capacity, customer satisfaction, and mood. The model uncovers interdependencies between the available parameters. Furthermore, it is able to simulate consequences of different preconditions and incidents during a sprint such as change requests.
In this contribution, we present our approach and apply it in a case study with three agile teams in industry. We build a System Dynamics model and use it for sprint simulations. Our analysis determined, e.g., the teams' productivity during the sprint and their workload each day. The simulation increased the teams' awareness of the negative influences due to interventions during the sprint.

Keywords: Process improvement · simulation · System Dynamics · agile software development teams · social aspects

1 Introduction

Companies strive for a good rate of successful projects. To reach this aim, they often face the necessity for software process improvement. There are various existing approaches to improve the development process [14]. However, it does

not necessarily suffice to apply the existing approaches. Implementing an agile environment is a commonly used possibility to overcome many problems. Hence, it is a common way to improve the process [14]. However, becoming agile is not always the key [8]. The same problem may occur when implementing other approaches to improve the development process. We expect that team members lack an understanding for the interdependencies within the entire software development process [13]. This lack can reduce the success when implementing existing approaches.

We assume that a simulation model can increase this understanding by simulating and visualizing the dependencies [13]. In this contribution, we present an objective, data-based approach for visualizing and discovering project dynamics and tendencies which may be too subtle to be found by humans or which are too obvious to recognize the major consequences. To facilitate the application of our technique, we base our approach on already existing or easily collectible data for example from tools like JIRA. Based on this data, we automatically calculate correlations and interdependencies between the variables like productivity (represented by finished story points), mood (represented by scales from psychology [20] or from a mood board with emoticons) or the capacity. These calculations result in a System Dynamics model simulating and visualizing the hypothetic state of a sprint with given preconditions. When a team is modeled with System Dynamics, that team can use the model to recognize factors hindering an optimal sprint or impediments that need to be resolved in order to be more successful.

In this paper, we present our approach and its application in a case study at Arvato SCM Solutions[3]. The company initiated a collaboration with the Leibniz University Hannover considering fundamentals about methods and toolchains. These fundamentals would reveal, e.g., the tendencies of sprint success criteria, customer satisfaction during a sprint, or factors influencing a positive impression by the client. The industry partner had already experienced general impediments such as changing the scope of a sprint which has negative effects to reach the sprint goal. During the kick-off of the collaboration, an interest in social interactions and dependencies in the teams arose. The Software Engineering Group at Leibniz University Hannover has some experiences in analyzing interactions in a team such as the FLOW method[4] for information flow analysis in developer teams or tools for the analysis of interactions in meetings [17]. Developers, manager, Scrum master and product owner[5] of the industry partner wanted to know if these tools can be applied in an agile environment and if they can help to get better results for a sprint if the teams know their social impediments.

[3] https://scm.arvato.com/en.html

[4] http://www.se.uni-hannover.de/pages/en:projekte_flow

[5] At Arvato SCM Solutions, the manager, Scrum master, and product owner build a triumvirate for a team. The manager is responsible for the people management and their personal evolution, the Scrum master is responsible for the processes in the agile environment and the product owner is responsible for the enhancements of the product and the functional stories.

Arvato SCM Solutions wanted to get a supportive tool helping a development team to identify its impediments to reach the goal of its sprints. In particular, such a tool would help Scrum master and product owner to realize which influences their behavior can have.

As a further requirement, the team members should be able to understand the outcome of the model, i.e. the result of the simulation needs to be reliable. Otherwise, the team may refuse the model.

Building such a model is not trivial. While some parameters such as productivity (in terms of finished story points) can be easily quantified, others such as mood cannot. Hence, we need to combine both qualitative and quantitative data. We can formulate our research question as follows:

RQ: *Is it possible to build a System Dynamics model including both qualitative and quantitative data with comprehensible, i.e. traceable, results?*

As a first step towards this aim, the researchers from Leibniz University Hannover analyzed three development teams at Arvato SCM Solutions. The teams had just started their agile transition to Scrum some months before the study began. One team was in sprint number 22, another in sprint number 13 and the third team in sprint number 8 (with a length of two weeks per sprint). The case study took place over a duration of 18 months starting with a data collection and ending with the presentation of the model and its application. The duration of the study was due to the explorative nature. Replications of this study will not take that much time.

The simulation of the sprint and the analysis with the System Dynamics model uncover i.a. the fact that the teams' productivity increases towards the end of a sprint. Some of the found issues can also be found in other teams. Nonetheless, the model does not adequately represent the process and the interdependencies between the variables in a team of another organization. Therefore, we describe our approach to build the model. This procedure can be applied to other teams with a different data set.

The paper is structured as follows: In the following section, we give an introduction to System Dynamics. Previous work related to our research topic is presented in Sec. 3. Section 4 gives an overview of the study we conducted at Arvato SCM Solutions. It also provides information on building the model. We discuss our results in Sec. 5 and conclude in Sec. 6.

2 Background: System Dynamics

System Dynamics was developed to understand the behavior of complex systems [3]. In the 1950s, Forrester [3] presented his approach for modeling different parameters in order to understand industrial processes. System Dynamics enables

holistic analyses and simulations of complex and dynamic behaviors. The simulation is based on a model taking into account relationships between various project parameters. Forrester [3] presented a strategy for identifying such relationships and for modeling and simulating the interdependencies. We use this approach to gain more profound insights into the interdependencies and the project's dynamics [4].

2.1 Causal Loop Diagrams

Causal loop diagrams visualize interrelations between different variables in a system [3]. An example is visualized in Fig. 1. *Mood* (1), *Productivity* and *Customer Satisfaction* are considered. A positive marked directed edge (2), a so-called **positive causal link**, denotes a positive relation, i.e. if customer satisfaction increases, mood also increases. It is also possible to visualize negative relationships, i.e. if one parameter increases, the other one decreases and vice versa. Such links are called **negative causal links**. One example for this relationship is the influence of unexpected incidents during the sprint on mood: The more incidents, the more dissatisfied developers, i.e. the lower the amount of positive affect.

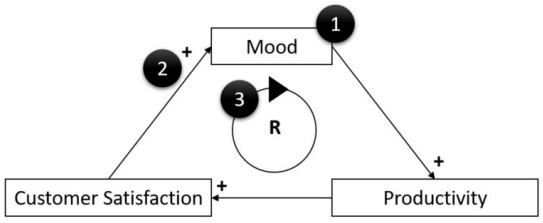


Fig. 1. Exemplary causal loop diagram visualizing a positive reinforcing loop.

A **reinforcing feedback loop** is defined to have an exponential increase or decrease. Mathematically, this is equivalent to having an even number of negative links (where 0 is also even). In Fig. 1, the reinforcing loop is visualized by (3). A **balancing feedback loop** is associated with reaching a plateau. This is equivalent to having an odd number of negative links. A balancing feedback loop is denoted as visualized with a "B" (instead of the "R" in (3)) and an arrow pointing counterclockwise [4].

2.2 Stock and Flow Diagram

Stock and flow diagrams concretize causal diagrams by quantifying the system. They are used to study and analyze systems on a more detailed level. An example is visualized in Fig. 2. These diagrams consist of **stocks** (1) and **flows** (3). A stock represents an entity, i.e. parameter, that changes over time. A flow defines

the rate of a change in a stock. So far unknown influences and parameters can be visualized by a **cloud** (2). In the given example, we have an influence of something not specified (2) on *mood* (1). The rate of the influence is defined by the flow (3).

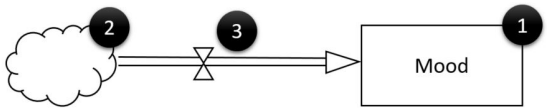


Fig. 2. Exemplary stock flow diagram.

Each flow can be concretized by an equation. These equations and the feedback loops enable a simulation of the process. The better the equations define the influences, the more accurate is the simulation in the end.

3 Related Work

This paper aims at analyzing interdependencies between different parameters during a sprint. Some of these parameters are **social aspects** like communication behavior [9, 10]. Since soft factors can be hardly quantified, the commonly used methods to analyze human beings are rather subjective. In this approach, we use already established methods for modeling and simulating the specific context, for instance with **System Dynamics** [3] to combine qualitative and quantitative data.

Klünder et al. [9] quantify communication behavior using the so-called FLOW distance. FLOW distance is a measure for indirections in information flow. Furthermore, the authors measure moods and social conflicts using scales which are established in psychology like the scale for positive and negative affect [20]. This way, the authors are able to find statistically significant relationships between communication behavior, mood, and social conflicts [9].

Herbsleb and Mockus [5] also investigate on communication behavior. They analyze the behavior of co-located and distributed work teams. Their study is based on quantified data from the source code change management and a survey. The authors aim at finding causes for delay in distributed teams. Herbsleb and Mockus [5] are able to uncover relationships between delay, communication, coordination, and geographical distance. They report a decrease in the frequency of communication with an increasing physical separation [5].

These studies base on subjective quantified data which often depends on the perceptions of the team members (cf. [9]). All proposed metrics we are able to measure are integrated into our approach.

To find interdependencies between qualitative and quantitative parameters as well as to integrate assumptions, we modeled the results using System Dynamics.

The idea of simulating human factors and other project parameters is not new. Abdel-Hamid and Madnick [1] investigated dynamic events in Software Engineering. They present different case studies of large software projects. Furthermore, they propose the use of metrics which help to measure the relevant aspects. Based on the data records and their experiences, the authors build system models with different complexities. Abdel-Hamid and Madnick [1] give a detailed overview of various dynamic models underlining, for example, the influence of productivity on the motivation of a development team. Furthermore, they present an entire software project process chain.

Cao et al. [2] investigate dynamics in agile software development. They model interdependencies between various agile methods and practices such as pair programming, customer involvement or refactoring and organizational parameters such as productivity or cost. The authors also use System Dynamics to generate the simulation model [2].

Madachy [15] models communication behavior and other team parameters. He simulates qualitative models to understand process dynamics. His work is based on formal boundary expressions with a wide range of parameter settings.

Hoegl et al. [6] present a concept about the quality of teamwork. They identify factors influencing the success of software projects. They consider aspects such as team performance and satisfaction. The relevant factors for teamwork are communication, coordination, a balance of contribution, mutual support, effort, and cohesion [6]. The authors base their results on an empirical study providing data from 575 developers and project leaders in 145 German software development laboratories [6].

Shiohama et al. [19] deal with the question of an appropriate iteration length. They present a procedure to calculate the recommended iteration length by simulating the sprint. They integrate parameters such as the development team, the probability of incidents during the sprint and the complexity of the project.

In previous studies, Kortum et al. [11] have already explored team behavior in academic software projects. The first models [7, 11] base on machine learning classifier and enable forecasts for key communication metrics. But these models only considered linear dependencies and few quantitative data. We want to extend these approaches by also considering non-linear interdependencies [13] and qualitative data.

In contrast to the already existing approaches, we do not want to answer a specific question using our simulation. Indeed, we want to explore team dynamics to uncover unexpected interdependencies. Nonetheless, we integrate results of already existing models such as measurements, metrics and best practices for setting up the model. Furthermore, we integrate data types which have – to the best of our knowledge – not yet been considered in related approaches. These are results from an interview study uncovering the information flow and results from a workshop representing subjective assumptions about the relationships to use unquantifiable data such as moods[6].

[6] There are possibilities to measure moods (e.g. with the PANAS scale [20]). But in our case study, it was not possible to measure moods retrospectively.

4 Study: Setting Up the Model

In order to examine the applicability of our approach, we conducted a case study in a company. The whole study based on a collaboration between Arvato SCM Solutions and the Leibniz University Hannover.

4.1 Methodology

Our general approach can be divided into several phases starting with the data collection and ending in the simulation model. An overview of these phases is visualized in Fig. 3.

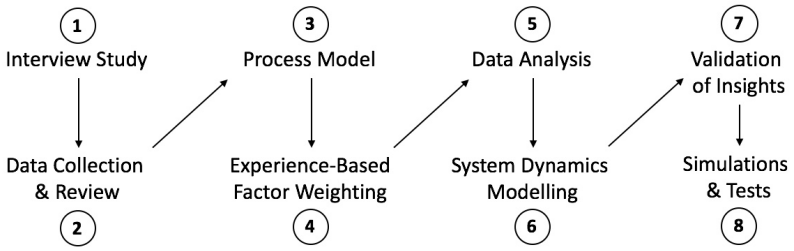


Fig. 3. Overview of the process

(1) Phase one starts with *interview studies* to get to know the culture of the company and the processes from an external viewpoint. We need to gain these insights in order to prepare the causality model. The structure of the interviews bases on the FLOW method [18] which is an already established proceeding in software engineering to get an overview the information flow within an organization and to get to know the process from different viewpoints.

(2) Afterwards, we collect different kinds of easily available *data records*. It is necessary to receive as much data as possible influencing the main issues which should be analyzed with the simulation model. Later, it is possible to integrate assumptions for non-accessible data (see (4)). We recommend using data for example contained in JIRA such as productivity, finished and open story points, sprint interruptions and so on.

(3) In phase three, we create a potential *process model* also visualizing assumed dependencies between the collected data and other variables like mood. This model does not necessarily rely on the data. Calculations are also not necessary. This first instance of the process model only represents possible and intuitive interdependencies between the variables. It will be validated later.

(4) During phase four, we integrate *subjective perceptions* from different team members, Scrum master, and other involved persons. They are asked to rate the causal effects in the process model (for example the influence of customer satisfaction on team mood) and to specify and concrete the model. The more

different persons the more reliable and the more complete the subjective results are. We recommend using a scale between -3 (for a strong negative relation) and +3 (for a strong positive relation). A rating of 0 indicates no relationship between the two variables. This way, the analyst gains an overview and a first idea of relationships and dependencies. This step is very important for the developers' awareness of the final model.

(5) Afterwards, the *analysis* starts with a visual preparation of the data with respect to history, temporal progressions and other effects. The analyst also looks for relationships between variables in the data set that are easy to detect. In the best case, these relationships also fit the interview study and the model resulting from step (4).

(6) Phase six aims at creating the *System Dynamics model*. This step is the most time-consuming one. A description on how to set up the model can be found in [12] and [13]. We transfer the causality model into the System Dynamics model. The relationships between different project parameters can be retrieved using *explorative data analysis* for example with MINE [16]. This analysis helps to detect dependencies between the variables. Furthermore, the explorative analysis can detect relationships which have been unknown so far. Afterwards, we need to formalize the dependencies in order to include them into the System Dynamics model. We need further analyses, logical associations and a clean-up of the data set with respect to missing data. In this step, we can use the ratings from step (4). For example: In the case that all developers state that the presence of the product owner during the sprint is very important for their motivation, this fact needs to be considered in the model even if there exist no underlying data. This way, we conclude the model step by step. It gets more and more concrete and detailed. The whole model might be too large. Hence, irrelevant factors (according to the aim of the model) can be identified and eliminated. The model's behavior in seldom situations is also very interesting. Hence, the model should be analyzed by running the simulation 1000 times.

(7) In the last step, we *validate the insights* from the model by comparing them with the data sources, i.e. the development teams. This can be done best during a workshop with the developers, the managers and the persons who are responsible for the process. Presenting some behavior and structures with the model may or may not lead to a "wow"-effect. The model aims at increasing the awareness of the involved persons with respect to possibly unknown or forgotten interdependencies. Hence, it is very good if the persons agree with the model.

(8) Now, the persons can start *using and testing the model* by changing some parameters like sprint interruptions. The more intuitively the model behaves, the more they will rely on it. But some unexpected issues are also important to show the impact of the model. It does not aim at forecasting exact values, but it aims at visualizing the system's behavior and increasing the transparency of consequences in order to enable some improvements.

4.2 Execution: Case Study in Industry

The whole study at Arvato SCM Solutions was performed as on-site research, where the researchers got to know the common behavior of managers, some Scrum master, and developers. By being involved in meetings with regular team members participating, the researchers could grasp a better understanding compared to an external person about the collaboration structures and company spirits. The collected impression and representative system data formed the fundamentals of the idea of a simulation model about the team's operational behavior in such this exemplary company.

(1) Some weeks later, the interview studies started according to the *FLOW method* [18]. A FLOW interview has a generalized process starting with some general questions like tasks within the team or the years of experiences of the interviewed person. This way, a basic understanding of the internal processes in the Scrum teams and the information exchange with the product owner and the customer emerged. For each task, the interviewee reports about the outcome as well as incoming, controlling and supporting issues like templates, knowledge or checklists [18]. The researchers interviewed three persons: one product owner, one developer and one SQL expert who is part of two teams. Descriptive data can be found in Table 1. The interviewees are members of the teams we considered in the further analysis. The team members were free to decide whether they wanted to give an interview or not.

Table 1. Overview of the interviewees and the duration of the interviews

	Interviewee I	Interviewee II	Interviewee III
Duration of the interview	2 hours	1,5 hours	2 hours
Team	1	2, 3	3
Team exists since	3 months	6 resp. 3 months	3 months
Team member since	3 months	6 resp. 3 months	3 months
Role	Product owner	SQL expert	Developer

(2) One researcher from Leibniz University Hannover dedicated two weeks on-site at Arvato SCM Solutions to gain deeper insight into communication behavior and social aspects such as mood, productivity and the collaboration between the team and the product owner. The researcher observed three teams who agreed to participate in the study. Data from JIRA have also been collected from all sprints such as burndown charts, i.e. the reduction of unfinished story cards during the sprint. The researchers received data about sprint results, i.e. finished and not-finished story points, and the burndown charts. Furthermore, data about the teams' capacities and the customer satisfaction which have been collected in the company right from the beginning have also been included in the model. Unfortunately, there was no data representing the mood of a single team, because the "mood board" is used by all teams. Hence, it only represents

the overall mood and it was impossible to distinguish between the data from the three observed teams and all the others in the company.

(3) Afterwards, the modeling phase started by drawing a causality diagram mainly resulting from the insights of the work shadowing. It only contains rather obvious dependencies and relationships. It was designed very light-weighted with some keywords like mood, motivation, productivity and capacity connected by directed arrows. The causality diagram is visualized in Fig. 4. The researchers expect an influence of HR-Capacity, i.e., the availability or absence of developers, on the sprint and the commitment constancy, i.e., the proportion of the story points the team wanted to finish during the sprint and the story points which are finished. This parameter influences customer satisfaction which influences the team mood. This, in turn, influences the whole sprint which also influences burndown charts and the customer satisfaction. The burndown chart represents the number of already finished story points and is hence an indicator of commitment constancy.

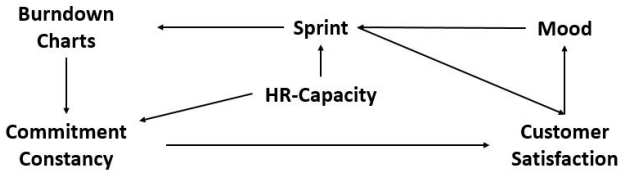


Fig. 4. First causality diagram in the end of step (3)

(4) The model in Fig. 4 was the basis for the next step. Two researchers met some interested persons from the company in a two-hour-workshop. Two Scrum master and four developers attended the workshop. One manager also temporarily attended the workshop. The participants were asked to rate the relevance of the relationships and find other dependencies. In pairs, they completed the diagram, removed arrows, added weightings representing the number of influences of one parameter on another one, and so on. We used a scale as proposed in the previous subsection in step (4). At the end of this workshop, the researchers had an overall causality diagram representing the perceptions of the persons who are involved in the process. This step was important to increase the credibility of the model. The cumulated results of this step are visualized in Fig. 5. Compared to Fig. 4, we have much more parameters after this step. Most of the parameters from step (3) remain, but the developers and the Scrum master stated a strong influence of meetings like refinements, the retrospective and the review on other factors. Furthermore, they rated the availability of the product owner as important.

(5) Afterwards, the data analysis started. The researchers used system records provided by JIRA and subjective responses about each sprints team capacity and the customer satisfaction. They derived metrics such as productivity as a

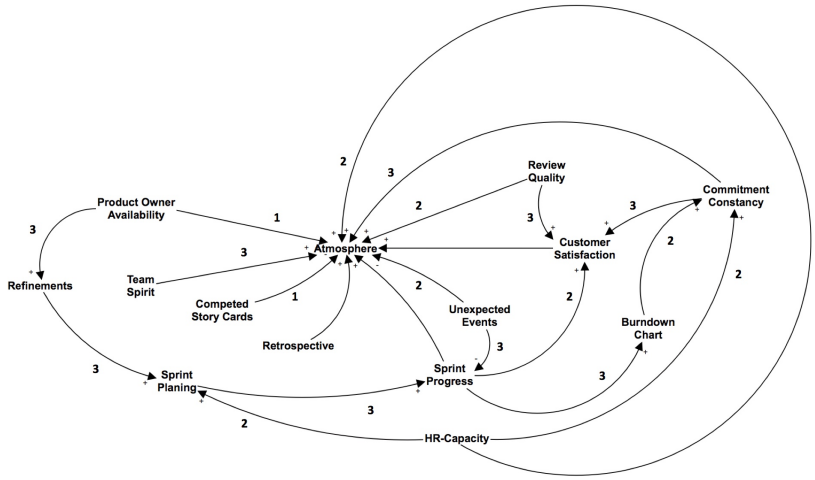


Fig. 5. Causality Model with weighted relationships (+/-: positive/negative relationship (i.e. "the more, the more" / "the more, the less") in the direction of the arrow; **1/2/3**: weak/medium/strong relationship)

measure defined via finished story points per hour or the customer satisfaction index derived from the customers' ratings. The metrics were combined for example with timely delivery and resource balancing. Additionally, timestamps have been divided into weekdays, holiday breaks and other events that could influence the teams' regular performances. Due to the information on burndown charts, i.e. the relation of finished and remaining story points, unplanned work (e.g. bug fixing) has also been considered. These incidents could be recognized due to an increasing number of story points in the sprint.

(6) In fact, all this information from the first five steps, especially the outcome presented in Fig. 5, allow to derive a first System Dynamics model based on the standardized stock-flow terminology [4]. In its very first building step, it only gives a system overview of all involved endogenous and exogenous components [4]. For the formalization of functional equations representing the interdependencies between various system components, we applied exploratory factor analysis to describe historical JIRA data in multi-functional equations. The dependencies and influencing factors without available objective data records become distinctively equalized through the rating from step (4).

The System Dynamics module for the productivity is visualized in Fig. 6 (marked with an (x)). This is one part of the whole System Dynamics model. Additionally, a dashboard user interface was built for exploratory simulation without a detailed need for knowing about every single module. A screenshot can be found in Fig. 7. The System Dynamics model and dashboard interface can be accessed and explored online[7].

[7] The System Dynamics model is available via http://www.goo.gl/Bnavhb.

Performing this step for each of the variables and combining each of the modules leads to the model presented in Fig. 6.

4.3 Verification and Validation

(7) For a better validation of the model's functional units, the researchers modularized each central and relevant metric in a separate function block. The correct functional operating was approached with real input ranges, whereas the outcome passed a plausibility test due to realistic operational ranges. Each functional module such as the one for productivity (marked with an (x)) in Fig. 6 characterizes its internal factor and dependencies and become solely verified in its input and outcome behavior according to experts expectations and data records from the previous sprint.

To gain objective results, we performed a sensitivity analysis to consider the influence of a given set of starting parameters for the simulation. Fig. 8 visualizes the daily productivity dynamics within one sprint. The x-axis represents the day of the sprint, and the y-axis covers the productivity on a rational scale. Since the model also takes weekends into account: The productivity is 0 some days. We have run 9000 sample simulations with the System Dynamics model and randomly generated parameter settings within the realistic operating ranges. As visualized in Fig. 8, the 100 variations of parametric inputs show that the common productivity follows a constant level for the first week of a new sprint. The results also sample that most dynamics during a sprint occur during the second half of a two weeks sprint. The course of the curves is comparable. Sensitivity analysis, in particular, involved the parameter inserts for a sprint planning about available human resources (60-150 working hours) and story points (80-120). There is an obvious increase in productivity in the last days of the sprint which seems to keep its proportional distribution regardless of whether teams have to face a high workload or not. In fact, the simulations pointed out that the teams typically finish larger story cards upfront the end of a sprint. Furthermore, they seem to finish story cards even with double speed compared to tasks at the beginning of the sprint. The term "double speed" is relative since the results also can be interpreted as that teams work slower at the beginning of a sprint and increase their regular performance shortly before the due date. In fact, the simulation uncovered some of the real situations that could be also confirmed by the management about their perceived impression on the team's typical workload during a two weeks sprint. When teams faced real pressuring situations due to last minute changes, customer claims, or an inappropriate sprint planning, the resulted team atmosphere became strongly affected, whereas particular situations could be also matched within the system model. This analysis based on a regular sprint with ten days and without extraordinary events decreasing a team's productivity such as holiday breaks. During sprints with breaks, for example, due to holidays, the peak of maximum productivity appears to be more present right after the holidays, whereas the productivity tends to reach its minimum right before the break.

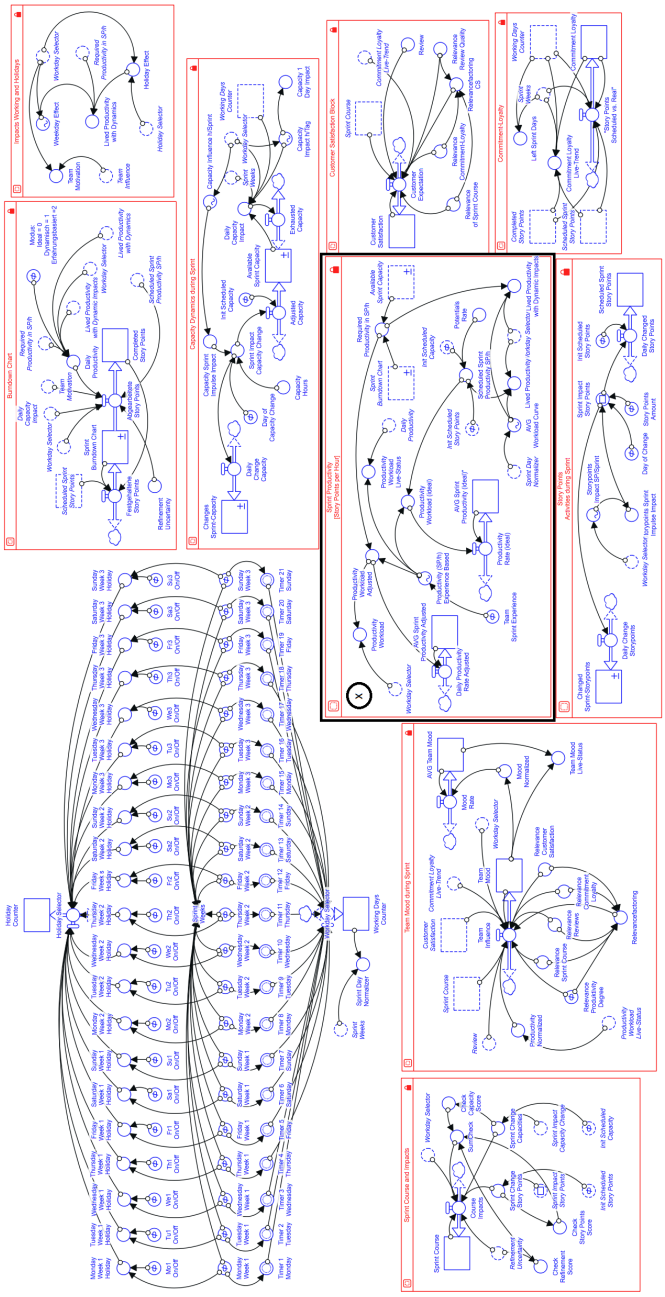


Fig. 6. System Dynamics Model. The productivity module is highlighted.

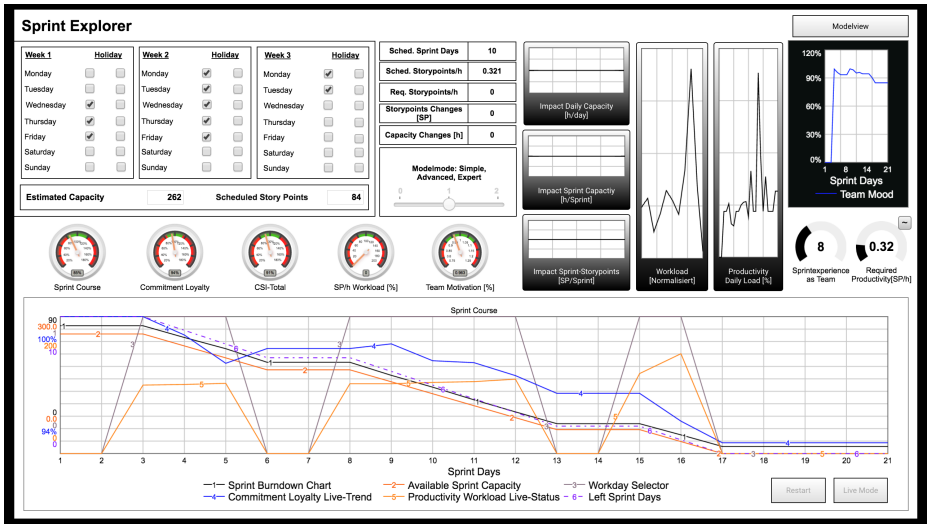


Fig. 7. Dashboard visualizing different sprint parameters after having entered given preconditions

4.4 Reliability of the Model

(8) To ensure the credibility of the entire model, some developers, Scrum master, product owner, and manager used it with different inputs and decided on the reliability of the results.[8] The conjunctions of all function blocks with the system resulted in statistical measures and subjective experience ratings as well. They were also validated through data records. The practitioners tested the model using different preconditions of sprints they had in mind. Some of the sprints have been "regular" ones, i.e. sprints without incidents and a satisfying sprint result in terms of story points, customer satisfaction etc. Others of the sprints used for testing the model have been outliers, i.e. with many incidents like bugs, fluctuations in the capacity due to illness and company-wide both pleasant and unpleasant events. According to the practitioners in the three observed teams at Arvato SCM Solutions, the model and the visualization with System Dynamics simulates the sprints of the three teams very well. The visualization shows similar sprint results in the simulation and as expected (in futurespective) and as it has been (in retrospective) in real life.

The curve of mood during a sprint is also near to reality according to the product owner and the developers. This was checked with sprints out of the period of data collection, including sprints after this phase and also with sprints in other teams in the company environment at Arvato SCM Solutions.

[8] For validity reasons and to test the generalizability (to some extent) of the model, we also asked team-external persons (but no company-external ones) to test the model.

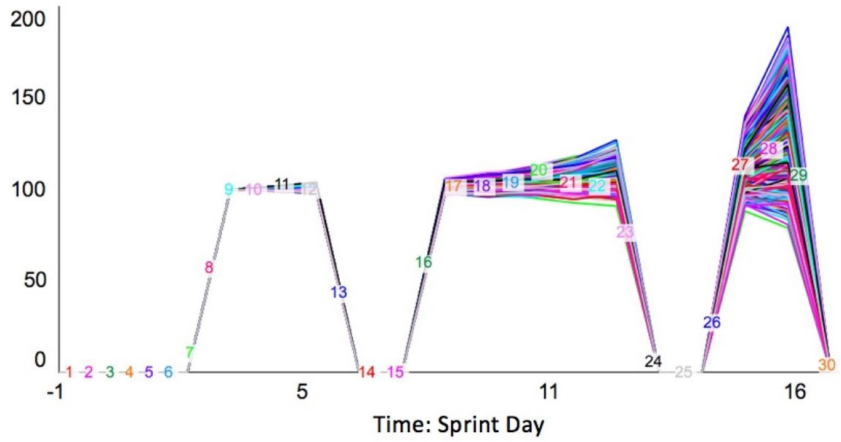


Fig. 8. Productivity variance results from 100 sample simulations

All in all, the model behaves intuitively in many points. This is very important for the reliability of the model. Nonetheless, there are some relationships which have been perceived to have a lower relevance for the development process.

4.5 Results and Implications

We, i.e. the researcher, manager, Scrum master, and other team members, detected some findings revealed by simulating real and potential sprints with the model. In Table 2, we present some of these weaknesses with suggestions for improvement. The suggestions and the causes for the findings are not complete. The causes need to be identified. This still requires manual effort and background knowledge. Combining the simulation with insights from the researcher's hospitation and the practitioners' experience led to the following result.

5 Discussion

According to our results, we can affirm our research question. It is possible to combine qualitative and quantitative data in a System Dynamics model and maintaining the credibility.

5.1 Threats to Validity

There are some aspects and limitations which may have to threaten the validity of our case study. According to Wohlin et al. [21], we categorize these aspects as *construct, internal, external* and *conclusion validity.*

Table 2. Findings (F) revealed by the System Dynamics model, possible causes (C) and suggestions (S)

ID	Description
F1	*In the end of a sprint, the productivity increases.*
C1	The increase of productivity may be caused by too large story cards, the developers cannot finish earlier.
S1	The developers need to break down the story cards into smaller ones.
C2	The developers may forget to change the state of a story card (in JIRA and on the physical board).
S2	It is important that both boards are synchronized (latest in the next daily). Furthermore, at least the state of a story card in JIRA needs to represent the actual state of a story card.
F2	*There are often unfinished story points in the end of a sprint.*
C3	The goal of the sprint in the number of story points was unachievable.
S3	Provide the possibility to learn from former sprints, e.g. how many story points have been finished
C4	There have been too many incidents during the sprint leading to less capacity.
S4.1	Try to keep the number of incidents (e.g. helping other teams) small.
S4.2	Get to know about planned support in other teams before the sprint starts.
C5	There have been undetected dependencies between story cards.
S5	Spend as much time as necessary with planning the next sprint, i.e. backlog refinement, Sprint planning etc.
F3	*Incidents influence the motivation, the mood and the productivity.*
S6	An incident goes along with adapting the plan. Often, the developers lose focus and realize that they cannot reach their sprint goal. This mostly leads to dissatisfaction.
C6	It is often impossible to have no incidents. But avoiding firefighting situations helps the team to keep focused.
F4	*Positive incidents such as story cards that are excluded from the sprint are not always good.*
C7	An exclusion of a story card creates the feeling of having much more time for the other story cards. Hence, the productivity decreases.
S7	The team should be aware of possible negative impacts of positive incidents. However, with an increasing agile maturity, the team will be satisfied with positive incidents without reducing the productivity.

Construct validity. The presented model based on both objective and subjective measures. The causality diagram completely depends on the perceptions of some team members and Scrum master. In the case where we do not have had any objective data for the simulation model, we indicated the dependencies and influences as supposed by the developers. This may have influenced the perceptions of the developers when seeing the overall model and its reliability.

Internal validity. In this contribution, we present the results of a case study with three agile working teams within one organization. We used the data from all teams simultaneously to create one overall diagram. Hence, we cannot make any statements with overall validity. The three teams do not represent the whole width of agile working teams. It is like a case study that generalizations are only limited possible.

Conclusion validity. The data analysis was completely computer-supported. It was based on the MINE[16] algorithm as presented in [12]. Hence, the analysis is objective and the results based on the data, too. But we do not have any objective data for mood. Since we wanted to include this parameter in the model, we used subjective perceptions of experienced team members and Scrum master reporting on the influences of mood on other parameters and vice versa. But this data is only little reliable.

External validity. The results may not be over-generalized. But the proceeding for the creation of the model may be applied to different kinds of teams, even working in different contexts and with other work-organizations like the V-model. Basing the model on the data of other teams or even on different kind of data (i.e., different variables) will surely lead to a different model. At this time, the model is team-specific. But including the data from other teams and extending the model will allow generalizations.

5.2 Limitations for the Use in Industry

The model has restrictions when a team has a high degree of agile maturity. It took more than a year to get the model and the visualization by System Dynamics. In between most of the teams improved their skills. They improved their technical skills, they improved the ability to handle impediments and interventions in the sprint, they improved estimations of the stories, and they improved the skills in team working, communication and other soft skills. In sum, they improved their level of agile maturity. With the new team constellations, the model is difficult to validate, because Arvato SCM Solutions does not collect all the data needed as a database for the model anymore. Because of the agile maturity, these teams do not need all kind of data for their continuous improvement process.

Now the teams can handle interventions and changes in a sprint very well, and they reached the defined goal of the sprint near to 100% defined as committed vs. achieved story points. Also, the product owner has high confidence in the development team and therefore the mood is good – mostly. Because of the combination of missing data and the stable mood, the model and the visualization do not work anymore for our teams – at least in the current version.

Retrospectively, Arvato SCM Solutions could identify the same tasks for improvement for a team through this simulation. For example, most of the stories in a sprint were set to state *"done"* during the last three days in a sprint. This could mean that the team is most productive only at the end of the sprint. But according to the experiences of the managers at Arvato SCM Solutions, in most cases, there are too many stories in a sprint with more than 13 story points for a story, which is the largest possible number of story points for a single story card. They are too big. The teams do not have a chance to finish this story earlier in a sprint. One solution is to increase the ability of the team to split bigger stories into smaller ones which require knowledge in story splitting.

A conclusion: The module and System Dynamics simulation is working very well for teams which are in the transition from an old work environment to an agile framework like Scrum. There is a need to collect a lot of data to use this model and tooling. But using a ticket system like JIRA, facilitates the selection. It is not easy to identify the correct actions for improvement. However, the simulation will support a team to make their challenges more visible. Finding the right solutions will depend on the team, and they have to try out different ways.

5.3 Interpretation

Software process improvement is an important topic for many teams and organizations. However, detecting the potential for improvement is not always easy since most processes are too complex to see the interdependencies "on the fly". Tools can support the detection of issues that may be improved.

Software development teams prefer support by tools and the management tends to use visualizations. Visualizations like burndown charts show the current state and the past progress of a sprint. This System Dynamics model can simulate the next sprint and visualize a sprint result under different circumstances such as added or deleted stories, added resources supporting the team, sick members or members of vacation, etc. Based on the results, the team and the Scrum master can identify the challenges and try different variants and find potential solutions.

This concrete model and the simulation are only helpful for teams when they use Scrum as a work environment and do not have a mixture of a classic work environment such as the waterfall model. The goal of a successful sprint should be nearby 100% of achieving the committed stories and story points. A permanent over- or under-commitment for a sprint does not help to get a stable team result. Two other preconditions for using this model is to have a stable team, and the sprints have to have a fixed length for a long time.

To use this model, it is important to have a good base of data and information. All data about the sprints (stories, story points and the states of the stories) should be in a ticket system like JIRA. The information about team capacity and mood factor should be tracked in parallel. When this data is available for a team, this model can be adopted by another organization.

In summary, this model can be a good addition to the general toolset for Scrum. By simulating future sprints and running sprints from the past with

different circumstances, the challenges and impediments in a team can be better analyzed – and supported by a tool. All these information can help a team and the Scrum master to identify the next actions to increase their agile maturity and in the end the productivity and profitability of a team.

A foresight: This model and its simulation is a good base for full-service tooling which can be used by each Scrum team. All needed information for the sprint (stories, story points, length of a sprint, etc.), the number of team members, a capacity planning of the team and other parameters for soft factors should be tracked in one tool. JIRA is a system which handles most of the information when it is used as a ticket system by the development team. It is also possible to develop add-ons in JIRA to put additional data into a sprint. So an add-on for JIRA bringing in all needed parameters to the general sprint data can be used to visualize sprints with a different view.

6 Conclusion

In this contribution, we presented strong synergies between different techniques for analyzing teams. We combined statistical and information flow analysis to build a System Dynamics model simulating interdependencies between various team parameters in agile software development. We extended quantitative data from JIRA by subjective ratings from the teams. This combination of objective and subjective statements on interdependencies helped to form the simulation model with a good prescription of dynamic team behavior during sprints.

The outcome of running the simulation fits the first expectations of both the researchers and the developers as well as Scrum master, product owner, and manager. Despite the model simulates the reality very well and is easy to use, the time for the realization needs to be shortened to replicate this study. Currently, the model is limited in the possibilities for the application. It may be adapted to other teams and organizations, but at the moment, it is only applicable to the development teams at Arvato SCM Solutions or comparable ones. But within the scope of application, the model represents the reality very fine-grained and reveals interdependencies which have not been observed before.

References

1. Abdel-Hamid, T., Madnick, S.E.: Software project dynamics: an integrated approach. Prentice-Hall, Inc. (1991)
2. Cao, L., Ramesh, B., Abdel-Hamid, T.: Modeling dynamics in agile software development. ACM Transactions on Management Information Systems (TMIS) **1**(1), 5 (2010)
3. Forrester, J.W.: World dynamics. Wright-Allen Press (1971)
4. Forrester, J.W.: System dynamics, systems thinking, and soft OR. System dynamics review **10**(2-3), 245–256 (1994)
5. Herbsleb, J.D., Mockus, A.: An empirical study of speed and communication in globally distributed software development. IEEE Transactions on software engineering **29**(6), 481–494 (2003)

6. Hoegl, M., Gemuenden, H.G.: Teamwork quality and the success of innovative projects: A theoretical concept and empirical evidence. Organization science **12**(4), 435–449 (2001)
7. Klünder, J., Karras, O., Kortum, F., Schneider, K.: Forecasting communication behavior in student software projects. In: Proceedings of the 12th International Conference on Predictive Models and Data Analytics in Software Engineering. ACM (2016). https://doi.org/10.1145/2972958.2972961
8. Klünder, J., Schmitt, A., Hohl, P., Schneider, K.: Fake news: Simply agile. Proceedings of the Conference on Projektmanagement und Vorgehensmodelle 2017 (2017)
9. Klünder, J., Schneider, K., Kortum, F., Straube, J., Handke, L., Kauffeld, S.: Communication in teams – An expression of social conflicts. In: Proceedings of the International Conference on Human-Centred Software Engineering. pp. 111–129. Springer (2016)
10. Klünder, J., Unger-Windeler, C., Kortum, F., Schneider, K.: Team meetings and their relevance for the software development process over time. In: Proceedings of Euromicro Conference on Software Engineering and Advanced Applications (2017)
11. Kortum, F., Klünder, J.: Early diagnostics on team communication: Experience-based forecasts on student software projects. In: Proceedings of the 10th International Conference on the Quality of Information and Communications Technology (QUATIC). pp. 166–171. IEEE (2016)
12. Kortum, F., Klünder, J., Schneider, K.: Characterizing relationships for system dynamics models supported by exploratory data analysis. In: Proceedings of the 29th International Conference on Software Engineering and Knowledge Engineering. KSI Research Inc (2017)
13. Kortum, F., Klünder, J., Schneider, K.: Don't underestimate the human factors! exploring team communication effects. In: Product-Focused Software Process Improvement: 18th International Conference, PROFES 2017, Innsbruck, Austria, Proceedings. Springer International Publishing (2017)
14. Kuhrmann, M., Diebold, P., Münch, J.: Software process improvement: A systematic mapping study on the state of the art. PeerJ Computer Science **2**, e62 (2016)
15. Madachy, R.J.: Software process dynamics. John Wiley & Sons (2007)
16. Reshef, D.N., Reshef, Y.A., Finucane, H.K., Grossman, S.R., McVean, G., Turnbaugh, P.J., Lander, E.S., Mitzenmacher, M., Sabeti, P.C.: Detecting novel associations in large data sets. science **334**(6062), 1518–1524 (2011)
17. Schneider, K., Klünder, J., Kortum, F., Handke, L., Straube, J., Kauffeld, S.: Positive affect through interactions in meetings: The role of proactive and supportive statements. Journal of Systems and Software **143**, 59–70 (2018)
18. Schneider, K., Stapel, K., Knauss, E.: Beyond documents: visualizing informal communication. In: Requirements Engineering Visualization, 2008. REV'08. pp. 31–40. IEEE (2008)
19. Shiohama, R., Washizaki, H., Kuboaki, S., Sakamoto, K., Fukazawa, Y.: Estimate of the appropriate iteration length in agile development by conducting simulation. In: Agile Conference (AGILE), 2012. pp. 41–50. IEEE (2012)
20. Watson, D., Clark, L.A., Tellegen, A.: Development and validation of brief measures of positive and negative affect: The PANAS scales. Journal of personality and social psychology **54**(6), p. 1063 (1988)
21. Wohlin, C., Runeson, P., Höst, M., Ohlsson, M.C., Regnell, B., Wesslén, A.: Experimentation in software engineering. Springer Science & Business Media (2012)