The Declarative Approach to Business Process Execution: An Empirical Test

Barbara Weber[1], Hajo A. Reijers[2], Stefan Zugal[1], and Werner Wild[3]

[1] Quality Engineering Research Group, University of Innsbruck, Austria
{Barbara.Weber,Stefan.Zugal}@uibk.ac.at
[2] School of Industrial Engineering, Eindhoven Univ. of Technology, The Netherlands
H.A.Reijers@tue.nl
[3] Evolution Consulting, Innsbruck, Austria
Werner.Wild@evolution.at

Abstract. Declarative approaches have been proposed to counter the limited flexibility of the traditional imperative modeling paradigm, but little empirical insights are available into their actual strengths and usage. In particular, it is unclear whether end-users are really capable of adjusting a particular plan to execute a business process when using a declarative approach. Our paper addresses this knowledge gap by describing the design, execution, and results of a controlled experiment in which varying levels of constraints are imposed on the way a group of subjects can execute a process. The results suggest that our subjects can effectively deal with increased levels of constraints when relying on a declarative approach. This outcome supports the viability of this approach, justifying its further development and application.

1 Introduction

In today's dynamic business environment the economic success of an enterprise depends on its ability to react to various changes, like shifts in customers' attitudes or the introduction of new laws [1]. Process-aware information systems (PAISs) offer a promising perspective on shaping this capability, resulting in a growing interest to align information systems in a process-oriented way [2]. Yet, a critical success factor in applying a PAIS is that it can flexibly deal with process changes [3]. To address the need for flexible PAISs, competing paradigms enabling process changes and process flexibility have been developed, e.g., adaptive processes [4], case handling [5], declarative processes [6], and late binding and modeling [7] – for an overview see [8]. All of these approaches relax the strict separation of build-time (i.e., modeling or planning) and run-time (i.e., execution), which is typical for plan-driven planning approaches as realized in traditional workflow management systems (cf. Fig. 1). By closely interweaving planning and execution the above mentioned approaches allow for a more agile way of planning. In particular, users are empowered to defer decisions regarding the exact control-flow to run-time, when more information is available.

P. van Eck, J. Gordijn, and R. Wieringa (Eds.): CAiSE 2009, LNCS 5565, pp. 470–485, 2009.

Depending on the concrete approach, planning and execution are interweaved to different degrees, resulting in different levels of decision deferral. The highest degree of decision deferral is fostered by *Late Composition* [8] (e.g., as enabled through a declarative approach) which describes activities that can be performed as well as constraints prohibiting undesired behavior. An example of a constraint in an aviation process would be that crew duty times cannot exceed a predefined threshold. A declarative approach, therefore, seems to be particularly promising to support highly dynamic processes [6,9]. The support for partial workflows [9] allowing users to defer decisions to run-time [8], the absence of over-specification [6], and more maneuvering room for end users [6] are all advantages commonly attributed to declarative processes. Although the benefits of declarative approaches seem rather evident, such approaches are not widely adopted yet in practice. In addition, there is a lack of empirical evidence on how well declarative approaches perform in real-world settings. In particular, it is unclear how well users can cope with the gained flexibility provided by declarative approaches, especially when processes and their context become rather complex.

The goal of this paper is to pick up on the demand for more empirical insights into the use of declarative approaches. Specifically, we aim to investigate how different levels of constraints may impede on the success that end users will have using a declarative approach for handling a particular business case (i.e., process instance). While it might be expected that the declarative approach will be effective to deal with business cases when few constraints apply, it is not clear whether end users are capable of translating a large number of constraints into effective updates on their initial plans. Because all constraints must be satisfied, one can argue that the sheer number of constraints will obscure from an end user's view what proper actions are still available. But proponents of declarative approaches to process planning and execution expect that end users will have little difficulty in doing this. In fact, this would exactly be one of its strengths [10,11,12], although this has not been established yet in an empirical setting.

To test whether end users can indeed deal with constraints, we conducted a controlled experiment with 41 students from both the Eindhoven Univ. of Technology and the Univ. of Innsbruck. In that test, we have been particularly interested in the potential impact of such constraints on the *effectiveness* of executing a process. This paper reports on the results of what we assume to be the first empirical work testing the declarative modeling paradigm. The structure of this paper is as follows. Section 2 provides backgrounds, while Section 3 describes our experimental framework. Section 4 covers the execution and results of our experiment. Finally, Section 5 discusses related work, followed by a conclusion.

2 Background

This section provides background information on planning approaches, presents different techniques for decision deferral, introduces declarative processes as well as the software we used for our experiment, the Alaska Simulator.

2.1 Planning Approaches

The flexibility of existing PAISs is significantly influenced by the underlying planning approach. In the following, we differentiate between *plan-driven*, *agile* and *chaotic* planning approaches, each of which assumes a completely different view on planning. Both plan-driven and agile approaches consider planning to be an essential activity, while chaotic approaches often lack a sufficient degree of planning and regard plans as unnecessary paperwork. In plan-driven approaches, the planning is usually done at the beginning and is not a repeated effort like in agile approaches (cf. Fig. 1). Although in both plan-driven and agile approaches the value of planning is appreciated, they have entirely different perceptions of a plan. In the former a plan is viewed as a schema for execution. Uncertainty is addressed by carefully planning everything upfront, which is appropriate for highly predictable processes. In contrast, agile approaches use a plan more like a guideline supporting decision making and recognize that in dynamic environments plans are quickly outdated and become inaccurate [13]. Decisions are made at the last responsible moment, when most information is available [1].

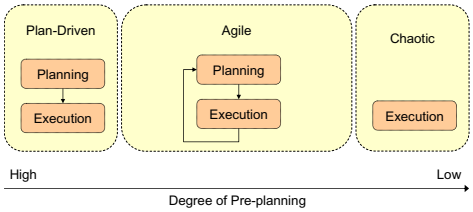

Fig. 1. Different Planning Approaches

2.2 Dealing with Uncertainty by Deferring Decisions

When examining existing PAISs, four different patterns for deferring decisions to the last responsible moment can be identified [8]. The *Multi-Instance Activity* pattern offers the least amount of freedom during run-time. It allows users to defer the decision on how often a specific activity should be executed to run-time, while the activity itself needs to be predefined. The *Late Binding* pattern offers slightly more flexibility by deferring the selection of the implementation of a particular process activity to run-time. Prior to execution, only a placeholder activity has to be provided; the concrete implementation is selected during run-time from a set of predefined fragments. The *Late Modeling* pattern goes one step beyond this and allows for the modeling of selected parts of the process schema at run-time. Prior to execution, a placeholder activity as well as a set of modeling constraints has to be defined. Most flexibility is offered by the *Late Composition* pattern, which allows users to compose process fragments from the process repository on the fly. No predefined model is required, as the business case can be created in an ad-hoc way by selecting activities from a repository,

while respecting all existing constraints. Consequently, *Late Composition* allows users to freely switch between process modeling and execution. The focus of this paper will be on *Late Composition* as enabled by declarative processes, which allows for the maximum level of flexibility.

2.3 Declarative Processes

There is a long tradition of modeling business processes in an *imperative* way. Process modeling languages supporting this paradigm, like BPMN, BPEL and UML Activity Diagrams, are widely used. Recently, *declarative* approaches have received increased interest and suggest a fundamentally different way of describing business processes [14]. While imperative models specify exactly how things have to be done, declarative approaches only focus on the logic that governs the interplay of actions in the process by describing (1) the activities that can be performed, as well as (2) constraints prohibiting undesired behavior. Imperative models take an 'inside-to-outside' approach by requiring all execution alternatives to be explicitly specified in the model. Declarative models, in turn, take an 'outside-to-inside' approach: constraints implicitly specify execution alternatives as all alternatives have to satisfy the constraints [15]. Adding additional constraints means discarding some execution alternatives (cf. Fig. 2). This results in a coarse up-front specification of a process, which can then be refined iteratively during run-time. Typical constraints described in literature can be roughly divided into three classes (e.g., [7,14]): constraints restricting the *selection* of activities (e.g., the minimum or maximum occurrence of activities, mutual exclusion, co-requisite), the *ordering* of activities (e.g., pre-requisite or response constraints) or the use of *resources* (e.g., execution time of activities, time difference between activities, budget, etc.).

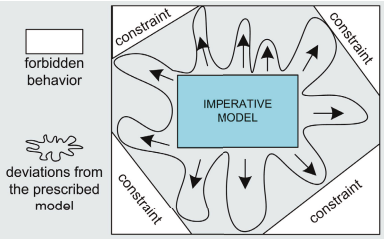

Fig. 2. Declarative Approaches to Process Modeling [11]

2.4 The Alaska Simulator

To foster the comparison of different approaches for process flexibility the Alaska Simulator[1] has been implemented, which takes a *journey* as metaphor for a *business process*. The similarities being exploited here are that regardless whether

[1] Developed at the University of Innsbruck, http://www.alaskasimulator.org

a journey or a business process is executed, various steps must be planned and carried out, even if the actual execution of those steps may be different from what is initially foreseen. Furthermore, journey planning is an attractive context for many people to become engaged in, which highly improves their willingness to use the system for experimental purposes.

In the Alaska Simulator, a plan can either be created in a plan-driven or in a more agile way (cf. Section 2.1). The Alaska Simulator also provides support for *Late Composition* as enabled by declarative processes. The actions of a journey, like travel activities, routes and overnight stays correspond to activities in the business process. For optimizing the execution of a particular business case, information about benefits (i.e., business value), cost and duration of activities is essential. Incomplete information prior to execution is a characteristic of both journeys and highly flexible business processes and is best handled by waiting until more information is available (cf. Section 2.2). The business value of a travel activity is not predefined, but can vary depending on the weather conditions during the journey. Thereby, the degree of variation depends on the activity's reliability. When composing a concrete business case, different constraints like *selection constraints*, *ordering constraints* or *resource constraints* have to be considered (cf. Section 2.3), similar constraints also exist when planning a journey (e.g., mandatory activities, dependencies between activities).

Fig. 3 depicts the graphical user interface of the Alaska Simulator. Users can compose their individual travel plan by dragging available actions from the Available Actions View (3) onto the travel itinerary (1). Actions are usually only available at a particular location on the map (4). Existing constraints are

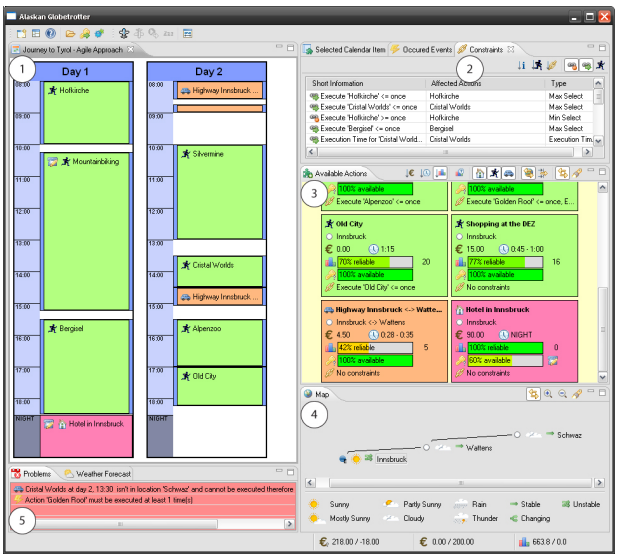

Fig. 3. Screenshot of the Alaska Simulator

displayed in the Constraint View (2) and have to be considered when composing a concrete journey. After each user action, the journey is validated and the user is informed about any constraint violations and plan inconsistencies(5).

3 Experiment Definition and Planning

The main goal of our experiment is to evaluate the outcome of end-users executing a business process that is guided by a declarative approach under varying numbers of constraints. This section explains the setup of our experiment (cf. Section 3.1) and introduces its specific design (cf. Section 3.2). Factors threatening the validity of the experiment results, as well as potential mitigations, are discussed in Section 3.3. We follow the recommendations given in [16] in setting up and describing an experiment throughout this section.

3.1 Experiment Setup

This section describes the subjects, objects and selected variables of our experiment, introduces our hypothesis and presents the instrumentation and data collection procedure.

Subjects: Subjects are 25 students of a graduate course on Business Process Management at Eindhoven University of Technology and 16 students of a similar course at the University of Innsbruck.

Objects: The objects to be modeled and executed are two declarative process models representing two journeys that must be planned and executed (referred to as Configuration Alaska and Configuration California). These configurations comprise activities to be executed, constraints that restrict their execution as well as their ordering, and events that might occur during run-time. For example, between a `Diving` activity and a `Flightseeing` activity there must be a rest period of two days to prevent aeroembolism. For each of the configurations, two variants are created: A and B, differing in the number of constraints only. To be specific, variant A contains a true subset of the constraints of variant B. An overview of the different variant characteristics is given in Fig. 4. Note that in addition to the constraints mentioned, in both variants all travel activities but one could be executed at most once, e.g., not visiting the Golden Gate bridge twice. To cover a broad and representative set of constraints we ensured that both configurations comprise constraints belonging to all three typical constraint classes (i.e., selection, ordering and resource constraints).

Factor and Factor Levels: The number of constraints in the model is the considered factor with levels "low" and "high". Variant A of a configuration corresponds to factor level "low" and variant B to factor level "high".

Response Variable: The response variable is the *business value* the subjects achieve when executing the journey. Thereby, the maximum achievable business

Variants	Actions	Constraints	Detailed Constraints
Alaska A	26	2	1 budget constraint, 1 end-location constraint
Alaska B	26	12	1 budget constraint, 1 end-location constraint, 3 mandatory actions, 3 execution time constraints, 1 constraint requiring a minimum delay between two actions, 1 pre-requisite constraint, 1 mutual exclusion constraint, 1 constraint requiring A to be followed by n times B and a final C
California A	22	2	1 budget constraint, 1 end-location constraint
California B	22	10	1 budget constraint, 1 end-location constraint, 4 mandatory actions, 2 execution time constraints, 2 constraints requiring a minimum delay between two actions

Fig. 4. Characteristics of the Configuration Variants

value for each activity is known upfront, whereas the value actually gained depends on the weather conditions during the journey (in the upfront phase only statistical data about the weather and the degree to which the business value can vary are known)[2]. To ensure comparability of results, weather conditions are the same for each subject.

Hypothesis Formulation: In our experiment we investigate whether adding additional constraints has an influence on the response variable *business value*. Based on this the following hypothesis is derived:

- **Null hypothesis** $H_{0,1}$**:** There is no significant difference in the business values of configurations with a low and a high level of constraints.
- **Alternative hypothesis** $H_{1,1}$**:** There is a significant difference in the business values of configurations with a low and a high level of constraints.

Instrumentation: To precisely measure our response variable we have implemented a logging function in the Alaska Simulator, which automatically records the required data.

Data Analysis Procedure: For data analysis well-established statistical methods and standard metrics are applied (cf. Section 4.2 for details).

3.2 Experiment Design

Literature on software experiments provides various design guidelines for setting up an experiment (e.g., [17]). Considering these design criteria, we accomplish our experiment as a *balanced single factor* experiment with *repeated measurement* (cf. Fig. 5). Our experiment is denoted as single factor experiment, since it investigates the effects of one *factor* (i.e., number of constraints in a configuration) on a common *response variable* (e.g., business value). Our experiment design also allows us to analyze variations of a factor called *factor levels* (i.e., configurations with few and with many constraints). The response variable is

[2] A detailed description on how to calculate the business value can be found in the Alaska Simulator's documentation (http://www.alaskasimulator.org).

determined when the participants of the experiment (i.e., *subjects*) apply the factor or factor levels to an *object* (i.e., Configuration Alaska or Configuration California). We denote our experiment as *balanced*, as all factor levels are used by all participants of the experiment, i.e., each subject has to plan a journey with both few and many constraints. This enables *repeated measurements* and thus the collection of more precise data, since every subject generates data for every treated factor level. Generally, repeated measurements can be realized in different ways. We use a frequently applied variant which is based on two subsequent runs (cf. Fig. 5). During the first run, half of the subjects (referred to as Group 1) apply few constraints to the treated object, while the other half (referred to as Group 2) uses many constraints. After having completed the first run, the second run begins. During this second run each subject applies the factor level not treated so far to the object. In order to avoid learning effects we use two different configurations for the two runs. Each subject was randomly assigned to either Group 1 or Group 2.

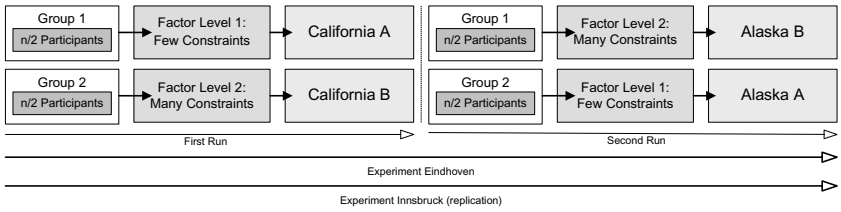

Fig. 5. Design of our Balanced Single Factor Experiment

3.3 Risk Analysis and Mitigations

When accomplishing experimental research related risks have to be taken into account as well. Generally, factors exist that threaten the internal validity ("Are the claims we made about our measurements correct?"), as well as the external validity ("Can the claims we made be generalized?") of an experiment. In our context, threats to *internal validity* are:

People: The students participating in our experiment differ in their skills and productivity. In particular, different experience levels in terms of process modeling, planning and scheduling may have an influence on the students' performance. This issue can only be balanced by conducting the experiment with a sufficiently large and representative set of students and to perform replications of the experiment. The number of 41 students seems sufficiently large to achieve such a balance. Furthermore, the experiment was replicated in the setting of the Innsbruck students after its initial conduct in the Eindhoven setting.

Besides, there are threats to the *external validity* of experiment results:

Students instead of professionals: Involving students instead of professionals can be critical. However, [18] has shown before that the results of student

experiments are transferable and can provide valuable insights into an analyzed problem domain. Moreover, for a journey planning and execution exercise as in this experiment, which requires no knowledge of a business domain, graduate students will probably have a similar ability as professionals.

Investigation of tools instead of concepts: In our experiment, the Alaska Simulator was used as a representative for a tool providing modeling and execution support for declarative business processes. Obviously, the achieved results to some degree depend on the quality of the used tool. Problems in understanding the tool as well as poor user support might influence the results. To mitigate this risk, considerable effort was put into designing an intuitive and easy to use user interface. As the Alaska Simulator was used at the 2008 Austrian Research Night by around 300 lay users of all ages (i.e., 7 to 70) without any considerable problems, tool understandability is not a major issue. However, it has to be recognized that our results cannot be automatically transferred to tools with less elaborated user support.

Choice of object: To mitigate the risk that the two variants of a configuration (i.e., few and many constraints) do not differ enough in terms of complexity, we performed pre-tests with several subjects (who did not further participate in the experiment) and repeatedly refined the configurations based on their feedback. We involved test persons both with in-depth knowledge of the Alaska Simulator and novices.

4 Performing the Experiment

By now, the set-up of the experiment has been explained. Section 4.1 describes the preparation and execution of the experiment. Then, the analysis and interpretation of the experiment data is presented in Section 4.2. Finally, in Section 4.3, a discussion of the experiment results is provided.

4.1 Experimental Operation

Experimental Preparation: As part of the set-up of the intended experiment, we prepared two travel configurations, i.e., Configuration California and Configuration Alaska. For each of the configurations, two variants were created: A (few constraints) and B (many constraints). To ensure that each configuration is correct and can be executed in the available amount of time, we involved several persons with different backgrounds in its pre-tests. Based on their feedback, the configurations were refined in several iterations. Finally, we compiled a "starter kit" for each participant, consisting of a screencast explaining the main features of the simulator and a test configuration to casually explore.

Experimental Execution: The experiment was conducted at two distinct, subsequent events. The first event took place during October 2008 in Eindhoven, a replication was performed three weeks later in Innsbruck. Prior to the start of

the experiment, all students had to attend an introductory lecture to obtain an overview on declarative processes. During this lecture, we further informed them about the goals and rules of the experiment. Afterwards, each student received his/her "starter kit". Having watched the screencast and having gone through the test configuration, the actual experiment started. In the first run, the students had to model and execute Configuration California. Half of them were assigned to the A variant of the configuration (containing few constraints), the other half to the more complex B variant. For all students, a familiarization phase of 25 minutes was available, in which they could explore the configuration and gather relevant domain knowledge. After this phase, the students had another 20 minutes to plan and execute the journey exactly once, with the goal to optimize the business value of the journey. In the second run of the experiment, Configuration Alaska had to be executed and the variants for the groups were switched, i.e., the students who had worked on an A variant had to work on the B variant and vice versa. Again, the actual execution of the configuration to obtain a high business value was preceded by a familiarization phase of 25 minutes.

Data Validation: After having conducted the experiment, the logged data was analyzed. We discarded the data of one Eindhoven student as the journey could not be properly executed due to a bug in the software that occurred only for this student. In addition, we did not consider the data of one Innsbruck student who performed the wrong variant of Configuration Alaska (A instead of B) in the second run. Finally, data provided by 25 Eindhoven students and 16 Innsbruck students were used in our data analysis.

4.2 Data Analysis

In this section, we describe the analysis of the gathered data and interpret the obtained results.

Descriptive Analysis: Based on raw data from the log of the Alaska Simulator we calculated some descriptive statistics for the response variable *business value* (cf. Fig. 6). By analyzing Fig. 6 one can observe the following:

	N	Minimum	Maximum	Mean	Standard Deviation
Experiment Eindhoven					
California A	13	0	4778,59	3477,39	1298,44
California B	12	0	4956,12	2278,75	2083,20
Total	25	0	4956,12	2902,04	1790,41
Alaska A	12	0	7254,13	5147,90	1887,27
Alaska B	13	0	7580,83	5117,30	2441,38
Total	25	0	7580,83	5131,99	2147,77
Experiment Innsbruck					
California A	8	0	6473,07	4563,34	1997,49
California B	8	2356,34	5816,09	4571,13	1179,95
Total	16	0	6473,07	4567,26	1584,84
Alaska A	8	4966,54	8328,13	6620,14	1107,89
Alaska B	8	5650,97	9043,25	7409,52	1035,02
Total	16	4966,54	9043,25	7014,83	1113,05

Fig. 6. Descriptive Statistics for Response Variable 'Business Value'

- For the Eindhoven sample, the mean business value for Configuration California A is higher than that for Configuration California B.
- For the Eindhoven sample, the mean business values for Configurations Alaska A and Alaska B are rather similar.
- For the Innsbruck sample the mean business value for Configuration California A is rather similar to that for Configuration California B.
- For the Innsbruck sample the mean business value for Configuration Alaska B is higher that that for Configuration Alaska A.

The question is whether the noted differences are statistically significant.

Data Plausibility: We analyzed data plausibility based on *box-whisker-plot diagrams*, which visualize the distribution of a sample and particularly show outliers. The diagram takes the form of a box that spans the distance between the 25% quartile and the 75% quartile (the so called *interquartile range* – IQR) surrounding the median which splits the box into two parts. The "whiskers" are straight lines extending from the ends of the box, the length of a whisker is at most 1.5 times the interquartile range. All results outside the whiskers can be considered as outliers. Fig. 7A shows the outliers for the Eindhoven sample. For all configurations except for California B outliers exist with respect to the obtained business values. As can be seen in Fig. 7B, for the Innsbruck sample only outliers exist for Configuration California A, while all data from the remaining configurations lie within the boxed areas. At a first glance, the number of outliers may appear rather high. However, this result can be explained by the fact that journeys which were finished with constraint violations were assigned a business value of 0. When these values are not considered, only one outlier remains for the Eindhoven sample and no outliers for the Innsbruck sample. Thus, plausible data distributions seem to be in effect.

Testing for Differences in Business Value: To test for differences in business values, we compare the business values obtained by the subjects using

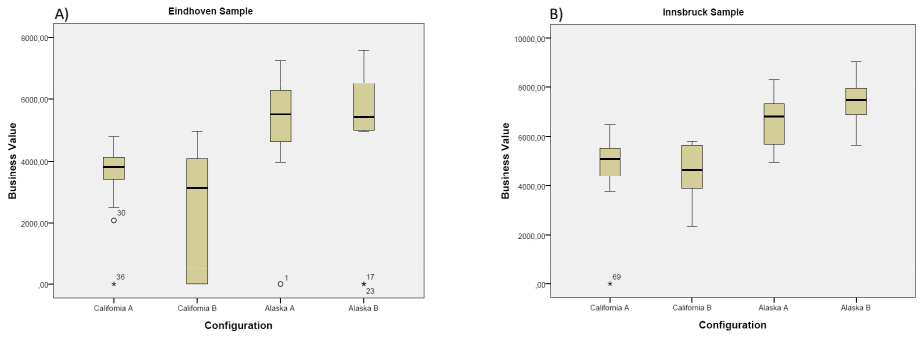

Fig. 7. Data Distribution (Box-Whisker-Plot Diagrams)

Configuration California A in the first run with those using California B. Furthermore, we compare the business values for the Alaska variants in the second run.

Eindhoven Experiment: Data for Configuration California A from the Eindhoven sample is not normally distributed, with standardized skewness and standardized kurtosis values outside the normal range. Therefore, a t-test cannot be applied to determine any differences in this case and the non-parametric Mann-Whitney test [19] – which can be thought of as comparing the medians of the distributions – is applied. With an obtained P-value of 0.125 (> 0.05), hypothesis $H_{0,1}$ cannot be rejected at a confidence level of 95%. Data for Configurations Alaska A and B for the Eindhoven sample is also not normally distributed. Thus, the non-parametric Mann-Whitney test is applied, resulting in a P-value of 0.643 (> 0.05). Like for the first test run, hypothesis $H_{0,1}$ cannot be rejected. So, for both Configuration Alaska and Configuration California there is no statistically significant difference between the business values obtained within their simple (A) and complex (B) variants.

Innsbruck Experiment: Data for California A from the Innsbruck sample is not normally distributed, thus a t-test is not applicable to determine any differences. Again, we apply the non-parametric Mann-Whitney test and obtain a P-value of 0.636 (> 0.05). Based on this result, hypothesis $H_{0,1}$ cannot be rejected at a confidence level of 95%. Data for Configurations Alaska A and Alaska B of the Innsbruck sample are both normally distributed and have the same variance. Therefore, we apply the t-test and obtain a P-value of 0.163 (> 0.05). Based on this, hypothesis $H_{0,1}$ cannot be rejected at a confidence level of 95%. The results of the replication confirm the results of the Eindhoven experiment. Again, for both the Alaska and the California configuration there is no statistically significant difference between the business values obtained within their simple (A) and complex (B) variants.

Overall Conclusion: Considering the results, strong support emerges for the conclusion that hypothesis $H_{0,1}$ cannot be rejected, i.e., no significant difference can be found between obtained business values in configurations with a low and a high level of constraints.

4.3 Discussion of Results

The major finding from our data analysis is that through our experiment no statistically significant differences can be found in the outcome of planning and executing a journey when we considerably vary the level of constraints that end-users have to take into account. This is supportive of the argument that end-users can use agile planning as enabled by a declarative approach to effectively deal with substantially varying levels of constraints.

The most plausible alternative explanation for the absence of any differences is that the configuration variants were not sufficiently distinguishable. We would

like to recall, however, that we refined the various configurations and their variants until the involved subjects in our pre-tests perceived a notable and considerable difference in difficulty between them (see Section 3.3).

Another alternative explanation, and a more technical one, is that the range of potential business values does not have enough spread to identify any differences. Yet, the coefficient of variation across the obtained business values for the various samples has ranged from 14% to 91%. This potentially offers sufficient variation to identify statistical differences, if there is any.

Considering that the alternative explanations do not appear stronger than the explanation we propose, i.e., the suitability and robustness of agile planning, the question that needs to be raised is to what extent our findings can be generalized.

First of all, planning a journey in the Alaska Simulator is not quite the same as collectively executing a business process. Furthermore, business processes can take on widely different forms and the required mix of support and flexibility may vary likewise. At the same time, the journey metaphor seems not to be a major threat to the validity of our results as similarities outweigh existing differences and the configurations used in the experiment range well beyond the size of toy examples, as they typically cover 22 to 26 actions to be planned and executed. It is clear to us that additional work is required to extend the scope and content of the experiment matter towards even more realistic settings. As is often the case, raising the level of external validity may be difficult without affecting the internal validity. In other words, it is questionable how the size and duration of a similar experiment could be extended to better reflect a realistic scenario without suffering from bad responses. Therefore, alternative empirical evaluations, such as gaming or case studies, may be more attractive instruments for further empirical research in this area.

Secondly, the Alaska Simulator provides its users means for creating a rough plan which is then incrementally validated. As such, users are well supported to become aware of constraint violations and resolve them. From a manual analysis of the planning actions (which were all logged during the experiment) we can establish that every subject created such a rough plan at the beginning. Based on some earlier experiences using the Alaska Simulator where subjects were unsuccessful when not following this approach, we suspect that the incremental validation of constraint violations and the ability to create a rough plan is essential for a good performance. So, it is unlikely that our results can be replicated by using a declarative system that does not provide this support, e.g., `DECLARE` [6], which is a clear restriction on generalizing our results.

5 Related Work

Most existing work about flexibly dealing with exceptions, changes, and uncertainty in the context of PAISs and related technologies is strongly *design-centered*, i.e., aiming at the development of tools, techniques, and methodologies. For overviews and discussions of these approaches, see [8,20,21].

Only few empirical investigations exist that aim to establish the suitability of the various proposed artifacts. In [22], the results of a controlled experiment comparing a traditional workflow management system and case-handling are described. The systems are compared with respect to their associated implementation and maintenance efforts. In turn, the impact of workflow technology on PAIS development and PAIS maintenance is investigated in [23]. However, these existing works primarily focus on traditional workflow technology, while this paper is the first one investigating the declarative modeling paradigm. Other empirical works with respect to PAISs mainly deal with establishing its contribution to business performance improvement, e.g. [24,25], and the way end-users appreciate such technologies, e.g. [26,27].

Worth mentioning here is a stream of research that relates to so-called *change patterns* [8]. It provides a framework for the qualitative comparison of existing flexibility approaches. This paper is complementary to that work by providing empirical findings in addition to the qualitative data presented earlier.

6 Summary and Outlook

Although the advantages attributed to declarative processes are manifold (e.g., support for partial workflows allowing users to defer decisions to run-time, the absence of over-specification as well as more room for end users to maneuver), their practical application is still limited. Furthermore, strengths and weaknesses of the declarative modeling paradigm are not yet well understood. In particular, it is unclear how well users can cope with the flexibility gained, especially when processes and their context become rather complex as the number of constraints increases. This paper reports on the results from what is presumably the first controlled experiment on the declarative process modeling paradigm. Our results indicate end-users can effectively use agile planning as enabled by a declarative approach over a considerable spectrum of constraints. However, incremental validation of constraint violations and the ability to create a rough plan seem essential ingredients for a good performance.

Our future work will aim at further investigating the practical suitability of declarative processes, in particular their maintainability. Although declarative workflows allow for easy changes of both business cases and process models by modifying constraints [6], it is notoriously difficult to determine which constraints have to be modified and then to test the newly adapted set of constraints. Acceptance testing, which is well established in software engineering, is known to facilitate communication regarding the intent of the developed software. To ensure that constraint changes are performed as intended, we consider transferring ideas from acceptance testing to the modeling of constraints.

In addition to this future line of research, we aim to investigate different techniques for improving understandability of declarative process models (e.g., through modularization or the definition of higher-level constraints) and we plan to validate our approach through further experiments.

Finally, we believe that there is a need for a benchmark to compare different declarative approaches. After all, it can be noted that different declarative approaches vary in respect to the extent that they hide procedural information from the modeler or the end user. It can be expected that results as we have reported in this paper may vary along that spectrum.

To conclude this paper, we wish to express our hope that the presented results will serve as an incentive for others to continue the promising development and application of declarative approaches for the planning and execution of business processes.

Acknowledgements. We thank G. Molina, M. Netjes, M. Song, J. Pinggera and T. Schrettl for their much appreciated help in preparing and executing the experiment.

References

1. Poppendieck, M., Poppendieck, T.: Implementing Lean Software Development: From Concept to Cash. Addison-Wesley, Reading (2006)
2. Weske, M.: Business Process Management: Concepts, Methods, Technology. Springer, Heidelberg (2007)
3. Lenz, R., Reichert, M.: IT Support for Healthcare Processes - Premises, Challenges, Perspectives. Data and Knowledge Engineering, 39–58 (2007)
4. Reichert, M., Dadam, P.: ADEPT$_{flex}$ – Supporting Dynamic Changes of Workflows Without Losing Control. JIIS 10, 93–129 (1998)
5. Van der Aalst, W., Weske, M., Grünbauer, D.: Case handling: A new paradigm for business process support. Data and Knowledge Engineering 53, 129–162 (2005)
6. Pesic, M., Schonenberg, M., Sidorova, N., van der Aalst, W.: Constraint-Based Workflow Models: Change Made Easy. In: Meersman, R., Tari, Z. (eds.) OTM 2007, Part I. LNCS, vol. 4803, pp. 77–94. Springer, Heidelberg (2007)
7. Sadiq, S., Sadiq, W., Orlowska, M.: A Framework for Constraint Specification and Validation in Flexible Workflows. Information Systems 30, 349–378 (2005)
8. Weber, B., Reichert, M., Rinderle-Ma, S.: Change patterns and change support features -enhancing flexibility in process-aware information systems. Data and Knoweldge Engineering, 438–466 (2008)
9. Wainer, J., Bezerra, F., Barthelmess, P.: Tucupi: a flexible workflow system based on overridable constraints. In: Handschuh, H., Hasan, M.A. (eds.) SAC 2004. LNCS, vol. 3357, pp. 498–502. Springer, Heidelberg (2004)
10. Hull, R., et al.: Declarative workflows that support easy modification and dynamic browsing. Software Engineering Notes 24, 69–78 (1999)
11. Pesic, M., van der Aalst, W.: A declarative approach for flexible business processes. In: Eder, J., Dustdar, S. (eds.) BPM Workshops 2006. LNCS, vol. 4103, pp. 169–180. Springer, Heidelberg (2006)
12. Mulyar, N., Pesic, M., van der Aalst, W., Peleg, M.: Declarative and procedural approaches for modelling clinical guidelines: Addressing flexibility issues. In: BPM 2007 International Workshops, pp. 335–364 (2008)
13. Cohn, M.: Agile Estimating and Planning. Prentice Hall Professional, Englewood Cliffs (2006)

14. van der Aalst, W., Pesic, M.: DecSerFlow: Towards a Truly Declarative Service Flow Language. Technical report, BPMcenter.org (2006)
15. Pesic, M.: Constraint-Based Workflow Management Systems: Shifting Control to Users. PhD thesis, Eindhoven University of Technology (2008), http://alexandria.tue.nl/extra2/200811543.pdf
16. Wohlin, C., Runeson, R., Halst, M., Ohlsson, M., Regnell, B., Wesslen, A.: Experimentation in Software Engineering: an Introduction. Kluwer, Dordrecht (2000)
17. Juristo, N., Moreno, A.M.: Basics of Software Engineering Experimentation. Springer, Heidelberg (2001)
18. Runeson, P.: Using students as experiment subjects - an analysis on graduate and freshmen student data. In: Proc. EASE 2003, pp. 95–102 (2003)
19. Siegel, S.: Nonparametric statistics for the behavioral sciences. McGraw-Hill, New York (1956)
20. Kammer, P., Bolcer, G., Taylor, R., Hitomi, A., Bergman, M.: Techniques for Supporting Dynamic and Adaptive Workflow. Computer Supported Cooperative Work (CSCW) 9(3), 269–292 (2000)
21. Reijers, H., Rigter, J., van der Aalst, W.: The case handling case. International Journal of Cooperative Information Systems 12, 365–391 (2003)
22. Mutschler, B., Weber, B., Reichert, M.: Workflow management versus case handling - results from a controlled software experiment. In: Proc. SAC 2008, pp. 82–89 (2008)
23. Kleiner, N.: Supporting usage–centered workflow design: Why and how?. In: Desel, J., Pernici, B., Weske, M. (eds.) BPM 2004. LNCS, vol. 3080, pp. 227–243. Springer, Heidelberg (2004)
24. Oba, M., Onoda, S., Komoda, N.: Evaluating the quantitative effects of workflow systems based on real case. In: Proc. HICSS 2000 (2000)
25. Reijers, H., van der Aalst, W.: The effectiveness of workflow management systems: Predictions and lessons learned. International Journal of Information Management 25, 458–472 (2005)
26. Bowers, J., Button, G., Sharrock, W.: Workflow from within and without: technology and cooperative work on the print industry shopfloor. In: Proc. CSCW 1995, pp. 51–66. Kluwer Academic Publishers, Dordrecht (1995)
27. Poelmans, S.: Workarounds and distributed viscosity in a workflow system: a case study. ACM SIGGROUP Bulletin 20, 11–12 (1999)