The Interplay of User-Centered Dialog Systems and AI Planning

Florian Nothdurft*, **Gregor Behnke**†, **Pascal Bercher**†,
Susanne Biundo† **and Wolfgang Minker***
*Institute of Communications Engineering, †Institute of Artificial Intelligence
Ulm University, Ulm, Germany
florian.nothdurft,gregor.behnke,pascal.bercher,
susanne.biundo,wolfgang.minker@uni-ulm.de

Abstract

Technical systems evolve from simple dedicated task solvers to cooperative and competent assistants, helping the user with increasingly complex and demanding tasks. For this, they may proactively take over some of the users responsibilities and help to find or reach a solution for the user's task at hand, using e.g., Artificial Intelligence (AI) Planning techniques. However, this intertwining of user-centered dialog and AI planning systems, often called mixed-initiative planning (MIP), does not only facilitate more intelligent and competent systems, but does also raise new questions related to the alignment of AI and human problem solving. In this paper, we describe our approach on integrating AI Planning techniques into a dialog system, explain reasons and effects of arising problems, and provide at the same time our solutions resulting in a coherent, user-friendly and efficient mixed-initiative system. Finally, we evaluate our MIP system and provide remarks on the use of explanations in MIP-related phenomena.

1 Introduction

Future intelligent assistance systems need to integrate cognitive capabilities to adequately support a user in a task at hand. Adding cognitive capabilities to a dialog system (DS) enables the user to solve increasingly complex and demanding tasks, as solving complex combinatorial problems can be delegated to the machine. Further, the user may be assisted in finding a solution in a more structured way. In this work we focus on the cognitive capability of problem solving via help of AI Planning technology in the form of Hierarchical Task Network (HTN) planning (Erol et al., 1994; Geier and Bercher, 2011). It resembles the human top-down way to solve problems. Such planners can help users to find courses of action, i.e., a sequence of actions, which achieve a given goal. In HTN planning the user states the goal in terms of a set of *abstract* actions, e.g., $train(abs)$, which are repeatedly refined into more concrete courses of action – using so-called methods – during the planning process. For example, $train(abs)$ could be refined into a $crunches(20)$ and a $sit\text{-}up(50)$ action. A solution is found if the the plan only contains *primitive* actions, i.e., actions which cannot be refined further, and the plan itself is executable.

In Section 2, we explain in more detail the advantages of integrating AI planning capabilities into a DS. Such an integration poses significant challenges, however. Most importantly, the way planners search for solution plans is very different from the way humans do, as their concern is mainly efficiency. In Section 3 we hence show how a planner can be adapted to better suit human needs. Further, we describe which kinds of planning-specific phenomena can not be avoided and thus the dialog manager must be able to handle. In Section 4 we describe the architecture of our intertwined system, followed by some remarks on the implementation in Section 5. Within this section we also discuss why a common source of knowledge for both the planner and the dialog manager is needed and how it can be created. Section 6 contains an evaluation of the implemented system in a fitness-training scenario.

2 Why Integrating a Planner?

In classical use-case scenarios for HTN planners (Nau et al., 2005; Biundo et al., 2011) a plan is generated without any user involvement, besides the specification of the goal, and afterwards presented to him. Hence, the planner is a black-box to the user, which is often not adequate. If executing the plan involves grave risks, e.g., in military

Proceedings of the SIGDIAL 2015 Conference, pages 344–353,
Prague, Czech Republic, 2-4 September 2015. ©2015 Association for Computational Linguistics

settings (Myers et al., 2002) or spaceflight (Ai-Chang et al., 2004), humans must have the final decision on which actions are to be contained in the plan. Planning systems can also be utilized to create plans for personal tasks like fitness training, cooking, or preparing a party. Here, it is expected that created plans are highly individualized, i.e., that they not only achieve given goals but also respect the user's wishes about the final plan. One might argue that such individualization could be achieved by integrating preferences or action costs into planning (Sohrabi et al., 2009). However, this approach requires that the user can specify his preferences completely and a priori and that they must be expressible, e.g., in terms of action or method costs or LTL formulae. Even if the user's preferences were expressible, it would be required to question the user extensively prior to the actual interaction, which is very likely to result in the user aborting the interaction.

Instead the dialog manager and especially the planner should learn about the user's preferences during interaction, fostering an understanding of the user's preferences. This requires the integration of the user into the planning process, resulting in a so-called mixed-initiative planning (MIP) system. A few approaches to creating such systems have already been investigated (Myers et al., 2003; Ai-Chang et al., 2004; Fernández-Olivares et al., 2006). Unfortunately, they cannot support a complete dialog with the user, as they mostly react on inquiries and demands from the user and only present the user with completed plans, i.e., plans that have already been refined into a solution. We deem such schemes impractical, as they require the user to comprehend a (potentially complicated) solution at once, making it hard to express opinions or wishes about it. For us, it would be more natural to iteratively integrate the user during the plan generation, making it on the one hand easier for the user to comprehend the plan and options to refine it, and on the other hand reducing the cognitive load, as the user does not have to understand the complete plan at once.

MIP can be interpreted as the system-initiated integration of the user in the planning process, but from a user's perspective it is the attempt to solve problems by using promised competencies of a technical system. For the user dedicating planning and decision-making to a technical system is done with the intent of finding a solution the user is not able to find at all or only with great effort. It aims at relieving the user's cognitive load and simplifying the problem at hand. Hence, the iterative integration of the user seems to be not only more natural, but also more practical for the user.

3 Challenges of MIP

In this section, we describe arising challenges of MIP. We discuss the differences between state-of-the-art AI Planning and the way humans solve problems, as they raise issues for a successful integration of a planner into a DS. To achieve it nevertheless, we show how a planner can be modified to accommodate them and which issues must be addressed by an advanced dialog management (DM).

How to Integrate the Planner. The integration of AI Planning begins with the statement of a planner objective in a dialog. This requires on the one hand a user-friendly and efficient objective-selection dialog, and on the other hand the creation of a valid *planning problem*. Thus, the semantics of the dialog has to be coherent to the *planning domain*, resulting in a valid *planning problem*.

User-friendly Search Strategies. Almost all AI Planning systems use efficient search strategies, like *A** or *greedy*, to find a solution for a given planning problem. The order with which plans are visited is based upon a heuristic estimating the number of modifications needed to refine the given plan into a solution. Plans with smaller heuristic value are hence regarded more promising and visited earlier. In A* search, as well as in any other heuristic-based search approach, it may happen that after one plan was explored, the next one explored will be *any* plan within the search space – not just one that is a direct successor of the plan explored last. As such, these strategies may result in the perception that the user's decisions only arbitrarily influence the planning process, which in turn may result in an experience of lack of control and transparency.

In contrast, humans tend to search for a plan by repeatedly refining the last one. A search strategy that resembles that strategy is *depth-first search* (DFS). Here, always the plan explored last is refined until a solution is found or the current plan is proved unsolvable, i.e., it cannot possibly be refined to a solution. In that case, the last refinement is reverted and another possible refinement option is chosen. If none exists the process is repeated recursively. A major drawback of DFS is that it does not consider a heuristic to select plans which are more promising, i.e., closer to a solution. DFS is blind, leading to a non-efficient search and non-optimal plans, i.e., the final plan may contain unnecessary actions. This problem can be addressed if the planner prefers refinements of plans with lower heuristic value if the user is indifferent about them. This scheme enables the interplay of user decisions and the planner's ability to solve complex combinatorial problems. The user controls the search until such a problem arises by selecting preferred refinements. Then he may signal the DS that he does not care about the remaining refinement, resulting in the planner using its heuristic to find a valid solution.

Handling of Failures During Planning. A major difference between human problem solving and DFS is the way failures are handled. In planning, a failure occurs if the current plan is proved unsolvable (e.g., by using well-informed heuristics). As mentioned earlier, DFS uses *backtracking* to systematically explore all remaining options for decisions that lead to a failure, until a successful option has been found. Practical heuristics are necessarily imperfect, i.e., they cannot determine for every plan whether it is unsolvable or it may be refined into a solution. Hence, even when using a well-informed heuristic, the planner will present the user with options for refining a plan that will inevitably cause *backtracking*. DFS *backtracking* is a very tedious and frustrating process, especially if the faulty decision was made early on but is found much later. Large parts of the search space, i.e., all plans that can be refined based on the faulty decision have to be explored manually until the actually faulty decision is reverted. This may result in the user deeming the system's strategy naive and the system itself incompetent. This is important, since the use of automation correlates highly to a user's trust into an automated system, which in turn depends mainly on the perception of its competence (Muir and Moray, 1996). To prevent the user, at least partially, from perceiving the system as incompetent, we can utilize the computing power of the planner to determine whether an option for refinement only leads to faulty plans. We call such refinements *dead-ends*. If options are presented to the user, the planner starts exploring the search space induced by each refinement using an efficient search procedure. If he determines that such a search space cannot contain a solution, the respective refinement is a *dead-end*. It is the objective of the dialog manager to appropriately convey this information to the user, especially if computing this information took noticeable time, i.e., the user has already considered the presented options and one has to be removed.

How to Integrate the User. Another important factor for a successful human-computer interaction is the question when a user should be involved into the planning process and if, how to do it. Clearly, the planner should not be responsible for this kind of decisions as it lacks necessary capabilities, but may contribute information for it, e.g., by determining how critical the current decision is with respect to the overall plan. From the planner's view every choice is delegated to the user via the DM, achieving maximal flexibility for the manager. The dialog manager on the other hand can either perform an interaction with the user, or determine by itself that the choice should be made by the planner, which is equivalent with the user signaling "Don't care". It should be considered whether interaction is critical and required to successfully continue the dialog or to achieve short-term goals, but risks the user's cooperativeness for interaction in the long run, e.g., by overstraining his cognitive capabilities or boring him. If the user is to be involved, the question arises how this should be rendered, i.e., what kind of integration is the most beneficial. Additionally, if he is not, the dialog manager must decide whether and if how he may be informed of the decisions the planner has made for him.

4 Concept and Design

The integration of AI Planning and user-centered dialog begins with the statement of an objective. This first dialog between user and machine has the goal of defining the task in a way understandable for the planner. Once the problem is passed to the planner the interactive planning itself may start. Using the described *depth-first search* the plan is refined by selecting appropriate modifications for open decisions. In order to decide whether to involve the user or not during this process, an elaborate decision model, integrating various information sources, is required. Relevant information sources are, e.g., the *dialog history* (e.g., was the user's decision the same for all past similar episodes?), the kind of *plan flaw* (e.g., is this flaw relevant for the user?), the *user profile* (e.g., does the user have the competencies for this decision?), or the current *situation* (e.g., is the current cognitive load of the user low enough for interaction?). Those examples of relevant information sources illustrate that a decision model can not be located *either* in the DM *or* the planner, but in a superordinate component, the so-called *Decision Model*.

In case of user involvement the information on the current *plan decision* has to be communicated to the user. This means that the open decision and the corresponding choice between available *modifications* have to be represented in a dialog suitable for the user. Hence, the corresponding plan information needs to be mapped to human-understandable dialog information. As this mapping is potentially required for every plan information and vice versa for every dialog information, coherent models between planner and DS become crucial for MIP systems. The thorough matching of both models would be an intricate and strenuous process, requiring constant maintenance, especially when models need to be updated. Thus, a more appropriate approach seems to be the automatic generation of the respective models using one mutual model as source, the *Mutual Knowledge Model*. This way, once the transformation functions work correctly, coherence is not an issue any more, even for updating the domain. How these essential constituents of a conceptual MIP system architecture (depicted in Figure 1) were implemented in our system, will be explained in the next Section.

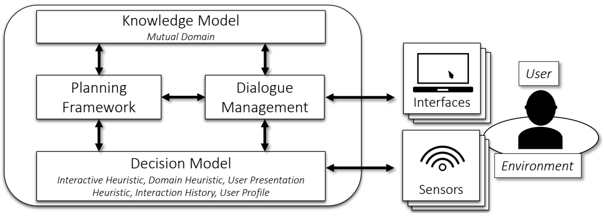


Figure 1: Essential components of a MIP system.

5 Implementation

We implemented and tested a multimodal MIP system using a knowledge-based cognitive architecture (Bercher et al., 2014). The multimodal interface uses speech and graphical user input as well as output. The *Dialog Management* uses a modality-independent representation, communicating with the user via the *Fission* (Honold et al., 2012), *User Interface* (Honold et al., 2013) , and *Fusion* (Schüssel et al., 2013) modules. Here, we will describe in more detail the two components, which are of particular interest for MIP systems: The *Mutual Knowledge Model* and the *Decision Model*.

5.1 Mutual Knowledge Model

This model, from which the planning and dialog domain models are automatically generated using automated reasoning, is crucial for a coherent MIP system. It is implemented in the form of an OWL ontology (W3C OWL Working Group, 2009). By generating the HTN planning domains from the ontology, a common vocabulary is ensured – for every planning task a corresponding concept exists in the ontology. Hierarchical structures (i.e., decomposition methods) inherent of HTN planning are derived using declarative background knowledge modeled in the ontology. For the delicacies of modeling planning domains in an ontology (e.g., how to model ordering or preconditions), including proofs for the transformation and remarks on the complexity of this problem, see Behnke et al. (2015) for further details.

The model is also utilized to infer a basic dialog structure, which is needed for the user to specify the objective for the planner. Using a mutual model addresses one of the challenges of MIP, since translation problems between dialog and planner semantics can be prevented. For the dialog domain generation a mapping between ontology concepts and dialogs is used. The dialog hierarchy can be derived using ontology knowledge. A dialog $\widetilde{A}$ can be decomposed into a sequence of sub dialogs containing the dialogs $\widetilde{B}_1, \ldots, \widetilde{B}_n$ by an axiom `Class:` A `EquivalentTo: includes onlysome` $[B_1, \ldots, B_n]$, which is interpreted by the dialog as a corresponding decomposition method. For example, a strength training can be conducted using a set of workouts $\widetilde{A}_1, \ldots, \widetilde{A}_n$, each of which consists of a set of exercises $\widetilde{B}_1, \ldots, \widetilde{B}_n$. This way a dialog hierarchy can be created, using the topmost elements as entry points for the dialog between user and machine. Nevertheless, this results only in a *valid* dialog structure, but not in a *most suitable* one for the individual user. For this, concepts of the ontology can be excluded from the domain generation or conjugated to other elements in a XML configuration file. This way unimportant elements can be hidden or rearranged for the user. The dialogs are also relevant during the MIP process. When selecting between several *Plan Modifications*, these have to be translated to a format understandable by the user. Hence, in addition to the knowledge used to generate plan steps, resources are required for communicating these steps to the user. Therefore, texts, pictures, or videos are needed, which can be easily referenced from an ontology. Using this information, dialogs suitable for a well-understandable human-computer interaction can be created and presented to the user.

One key aspect of state-of-the-art DS is the ability to individualize the ongoing dialog according to the user's needs, requirements, preferences, or history of interaction. Coupling the generation of the dialog domain to the ontology enables us to accomplish these requirements using ontological reasoning and explanation in various ways. The dialogs can be pruned using ontological reasoning according to the user's needs (e.g., "show only exercises which do not require gym access"), to the user's requirements (e.g., "show only beginner exercises") or adapted to the user's dialog history (e.g., "preselect exercises which were used the last time") and preferences (e.g., "present only exercises with dumbbells"). Additionally, integrating pro-active as well as requested explanations into the interaction is an important part of imparting used domain knowledge and clarifying system behavior. Using a coherent knowledge source to create dialog and planning domains enables us to use predefined declarative explanations (Nothdurft et al., 2014) together with dynamically generated plan explanation (Seegebarth et al., 2012) and explanations for ontological inferences (Schiller and Glimm, 2013) without dealing with inconsistency issues. This way *Plan Steps* (e.g., exercises) can be explained in detail, dependencies between plan steps can be explained to exemplify the necessity of tasks (i.e., plan explanation), and ontology explanations can justify decompositions from which the planning model and the dialog domain where generated. All of which increase the user's perceived system transparency.

5.2 Decision Model

This model is in charge of deciding when and how to involve the user in the planning process. It is the interface to the planner and decides, upon planner requests, whether a user involvement is useful. For this it includes a list of essential domain decisions that are interesting and relevant for the user (e.g., for a training domain: day, workout, and exercises) - the rest is left for the fallback-heuristic, and thus decided by the planner. Hence, the user is only involved in the decision making if a user-relevant planning decision is pending (e.g., "which leg exercise do you prefer?"). If it is in favor of user involvement the open decision and its modifications have to be passed to the user. Hence, the decision on the form of user integration has to be made. The dialog may either consist of the complete set of modifications, a pruned or sorted list, implicit or explicit confirmations of system-made preselections, or only of a user information. This decision depends not only on the interaction history, but also on additional information (e.g., affective user states like overextension, interest, or engagement) stored in the user state.

The *Decision Model* also records the dialog- and planning history. There are several reasons for that: The dialog history may enable a prediction of future user behavior (e.g., in selections), and additionally this knowledge is mandatory for *backtracking* processes, when the current plan does not lead to a solution. The history saves which decisions were made by the user. In case of *backtracking* the decisions are undone step-by-step, with the goal of finding a solution by applying alternative modifications. Whenever a user-made decision is undone, the user is notified, because this system behavior would otherwise appear irritating.

Since *backtracking* as well as *dead-ends* are peculiar phenomena in a MIP system, the communication of these might be a critical influence on the user experience. Together with the *DM*, the *Decision Model* orchestrates the corresponding system behavior. The main difference between *backtracking* and *dead-ends* is the temporal ordering of the awareness of the unsolvable plan and made decision. For *backtracking* the awareness is achieved after the decision, and for *dead-ends* during the decision. As we assumed that *backtracking* will impair the user experience significantly, a parallel search for *dead-ends*, as described in Section 3, was implemented. The process itself is, of course, inherently different from *backtracking*, but may prevent it. Removing dead-ends from the search space, when the relevant modification is not part of the current selection, is a rather easy task. Otherwise, the current selection has to be modified to prevent the user from selecting a *dead-end*. However, removing it without any notification from the list seems like a confusing behavior. As we had only hypotheses on the effects of these peculiar events as well as on the effects of the different forms of integrating the user into the planning process, we conducted a matching user study.

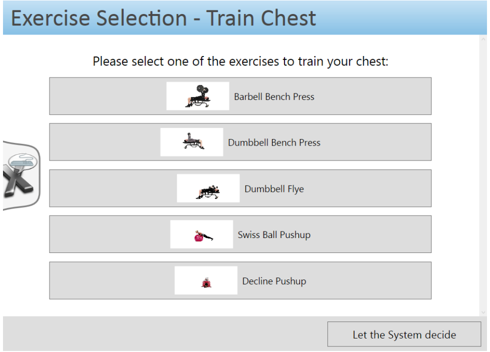
6 Evaluation

MIP may lead to the user experiencing planning-related phenomena, such as *backtracking* or *dead-ends*. These phenomena revoke decisions made by the user or alter the present decision-making and therefore may influence the user's experience of the system. As mentioned before, this may impair the perceived competency of the system, leading to a loss of trust, which correlates highly to a reduced use of automation (Muir and Moray, 1996). As our MIP system aims at assisting the user in complex and demanding tasks, maintaining the user's trust into the system and thereby the willingness to let the system decide autonomously is crucial. Furthermore, previous research has shown that the use of explanations can help to address trust issues related to intelligent adaptive systems (Glass et al., 2008) and that it may reduce negative effects in incomprehensible situations (Nothdurft et al., 2014). Therefore, we have assessed the effects of MIP phenomena like *backtracking* and *dead-ends* on the user-experience and tested different strategies, and especially explanations, to communicate these events to the user.

6.1 Methodology

For the subjective measurement through self-ratings by the user, questionnaires have been used. The most fundamental user data is personal information, assessing age, gender and education. We asked for the participants experience in the general use of technical systems and the user's foreknowledge in the corresponding domain of the experiment. Apart from the persona, we included a number of standardized and validated questionnaires: *Human-Computer Trust* (HCT) describes the trust relationship between human and machine and was assessed using the questionnaire by Madsen and Gregor (2000) measuring five dimensions (Perceived Understandability, Perceived Reliability, Perceived Technical Competence, Personal Attachment, Faith). The *AttrakDiff* questionnaire extends the assessment of a DS or software in general from the limited view of usability, which represents mostly pragmatic quality, to the integration of scales measuring hedonic qualities. This questionnaire was developed by Hassenzahl et al. (2003) and measures the perceived pragmatic quality, the hedonic qualities of stimulation and identity, and the attractiveness in general. In total 104 participants took part in the experiment. In average the subjects were 23.9 years old with the youngest being 18 and the oldest 41. Gender-wise the participants were almost equally distributed with 52.9% males and 47.1% females.

In this scenario the user's task was to create individual strength training workouts. In each strength training workout at least three different muscle groups had to be trained and exercises chosen accordingly. The user was guided through the process by the system, which provided a selection of exercises for training each specific muscle group necessary for the workout. For example, when planning a strength training for the upper body, the user had to select exercises to train the *chest* (see Figure 2). This selection corresponds to the integration of the user into the MIP process. The decision how to refine the task of training the chest is not made by the system, but left to the user.


Exercise Selection - Train Chest

Please select one of the exercises to train your chest:

Figure 2: Screenshot of the UI. Here, the user has to select one chest exercise. If he doesn't want to decide, "Let the System decide" can be clicked or said, starting the planner for this decision.

6.2 MIP Phenomena

For the MIP phenomena we implemented 4 variants: The variants used in the evaluation were the following: *Backtracking with Notification* (BT-N) where the system informs the user that previously user-made decisions will not lead to a solution and have to be undone. *Dead-End Notification before (DE-B)*, where the user was presented the notification, beforehand on an extra slide, that some refinements had to be deactivated in the upcoming selections, because they would not lead to a solution. *Dead-End Notification during (DE-D)*, where the selection provided, on each selection slide, the notification that some of the options below had to be deactivated, because they would not lead to a solution. The fourth variant tested the effect of the content of the notification. This means, that the *Backtracking with Explanation (BT-E)* variation additionally explained to the user why the already made decisions had to be rolled back. The participants were distributed by a random-function to the variants, resulting in 25 participants for BT-N, 18 for BT-E, 18 for DE-B, and 21 for DE-D (without unusable subjects).

6.2.1 Temporal Contiguity

Our first hypothesis was that in general the temporal contiguity of the system-provided notification does affect the user-experience. We assumed that providing the notification before the upcoming selection will perform better than on the selection itself, and way better than providing the notification after the decision, because the amount of consequences for the user is directly influenced by the temporal contiguity of the notification. In terms of HCT we expected that especially the bases of perceived reliability and understandability will be defected stronger by a larger temporal contiguity.

Results. There was a statistically significant difference between groups with notifications (i.e., without BT-E) in HCT-bases (see fig. 3) as determined by one-way ANOVA for *perceived reliability* ($F(2,70) = 3.548, p = .034$), *perceived understandability* ($F(2,70) = 4.391, p = .016$), and significant for *personal attachment* ($F(2,44) = 3.401, p = .042$). While analyzing the AttrakDiff questionnaire data we found a statistically significant difference for the dimension of *hedonic qualities - stimulation* between groups as determined by one-way ANOVA ($F(2,44) = 3.266, p = .048$). The Fisher LSD post hoc test revealed statistical difference at the .05 level between BT-N ($M = 4.04, SD = 1.03$) and DE-D ($M = 3.26, SD = .72$). Also DE-D was significantly different at the .05 level from DE-B ($M = 4.12, SD = 1.03$).

Discussion. The results support our hypothesis that the temporal contiguity of the notification does indeed influence the user-experience. A technical system will be perceived to be most reliable, when the notification is presented before the decision-making (DE-B), because no unexpected events occur, and the least when user decisions have to be undone (BT-N). For perceived understandability presenting the notification during the decision performed best, maybe because the deactivation of selection options could be allocated more directly to the notification itself and therefore foster the user's understanding of the situation. The personal attachment was mostly defected when using notifications during decision making. The results in general support that a positive temporal contiguity (i.e., DE-B) seems to be the best option for a technical system. While the understandability performs only second best, the perceived reliability, personal attachment, overall cognitive load, difficulty, fun, extraneous load and pragmatic as well as hedonic qualities, and overall attractiveness perform either best or as good as the other conditions using only notifications. This notification, which only represents some sort of shallow justification for the experienced system behavior, also seems to be important for the perceived user-experience. Hence, we evaluated how a real explanation would perform opposed to shallow justifications (i.e., the notification condition).

6.2.2 The Effects of Explaining MIP

For testing the effects of an extensive explanation, we exchanged the *backtracking* notification with an explanation. The notification that the made decision will not lead to a solution was exchanged with "the system has detected that the gym is closed today due to a severe water damage. Therefore, you have to decide again and select exercises suitable for training at home". This condition (BT-E) was then compared to the notification condition (BT-N). Thus, a pairwise t-test was used.

Results. Examining the HCT-bases we found significant differences between BT-N ($M = 2.8, SD = 1.05$) and BT-E ($M = 3.56, SD = .82$) for *perceived reliability* ($t(3.0) = 57, p = .004$). For *perceived understandability* the mean differed significant ($t(3.99) = 57, p = .000$) with BT-N ($M = 2.40, SD = 1.05$) and BT-E ($M = 3.44, SD = .87$). Observing the *perceived technical competence* BT-N ($M = 2.75, SD = 1.03$) and BT-E ($M = 3.28, SD = .66$) also performed significantly different ($t(2.06) = 41, p = .045$).

In the AttrakDiff we observed a significant difference ($t(2.37) = 41, p = .022$) for the dimension of experienced *pragmatic qual-*

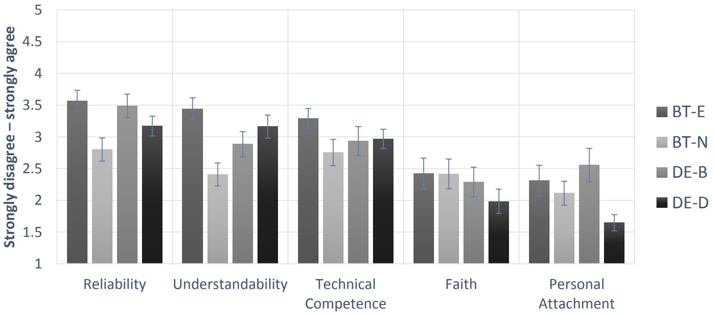


Figure 3: This shows the average mean of the bases of human-computer trust for each variant on a 5-point Likert scale. The whiskers represent the standard errors of the mean.

ities comparing BT-E ($M = 4.86, SD = 1.16$) and BT-N ($M = 4.01, SD = 1.15$). Taking a closer look at the word pairs significant differences below the .05 level were found for *complicated-simple*, *unpredictable-predictable*, *confusing-clearly structured*, *unruly-manageable*, as well as for *unpleasant-pleasant*.

Discussion. Providing detailed explanations of the system behavior, in this case of backtracking, does indeed help to perceive the system as more reliable, more understandable, and more technically competent. As only the *backtracking* notification was modified, we can only infer that for the other variants (i.e., DE-D and DE-B) the effect would be similar. However, this seems logical because the goal of increasing the system's transparency to the user can be achieved using similar explanations as well. Taking a look at the AttrakDiff and its single word pairs it becomes obvious that explaining system behavior helps to improve the pragmatic qualities of a system compared to providing none to minimal notifications. Systems with explanation capabilities seem to be perceived as not so complicated, more predictable, manageable, more clearly structured and in general as more pleasant.

Experiment Conclusion. Combing the results of both evaluated factors (i.e., temporal contiguity and explanations) we argue that the best option for MIP system behavior would be explaining the user why e.g., several options have been pruned from a selection beforehand. This strengthens the need for, on the one hand, intelligent and understandable explanation capabilities of such systems and on the other hand that the user is only integrated into the decision making when the system is sure that the presented options do in fact, or at least most probably, lead to a solution. Otherwise, the negative effects of occurring *backtracking* and similar planning peculiarities will impair the relationship between human and machine.

7 Conclusion

In this paper we pointed out the importance for future intelligent systems of intertwining dialog systems and AI Planning into a MIP system. First, we elucidated the potentials, but also the risks and arising problems of these mixed-initiative systems. On the one hand, humans can profit from planning techniques like parallel exploration, excluding non-valid planning paths from the search space. On the other hand, planning-related events like *backtracking* or *dead-ends* may impair the user experience. Second, we described our approach of a coherent and user-friendly mixed-initiative system. This included the use of a mutual knowledge model, in form of an ontology, to generate coherent domain models for dialog and planning as well as the development of a subordinate decision model, controlling who is in charge of the decision-making process. Furthermore, we evaluated our implementation on the effects of MIP events and tested different strategies to handle those. Concluding, we remark that the potentials of the integration of AI planning into a DS have to be weighed against the drawbacks like *backtracking* or *dead-ends* and their effects on the user experience. However, increasing the user's perceived system transparency by including valid explanations on these behaviors may mitigate the negative effects, thus increasing the potential areas of application for this kind of mixed-initiative systems.

Acknowledgment

This work was supported by the Transregional Collaborative Research Centre SFB/TRR 62 “Companion-Technology for Cognitive Technical Systems” which is funded by the German Research Foundation (DFG).

References

Mitchell Ai-Chang, John Bresina, Len Charest, Adam Chase, JC-J Hsu, Ari Jonsson, Bob Kanefsky, Paul Morris, Kanna Rajan, Jeffrey Yglesias, et al. 2004. Mapgen: mixed-initiative planning and scheduling for the mars exploration rover mission. *Intelligent Systems, IEEE*, 19(1):8–12.

Gregor Behnke, Denis Ponomaryov, Marvin Schiller, Pascal Bercher, Florian Nothdurft, Birte Glimm, and Susanne Biundo. 2015. Coherence across components in cognitive systems – one ontology to rule them all. In *Proc. of the 25th Int. Joint Conf. on Artificial Intelligence (IJCAI 2015)*. AAAI Press.

Pascal Bercher, Susanne Biundo, Thomas Geier, Thilo Hoernle, Florian Nothdurft, Felix Richter, and Bernd Schattenberg. 2014. Plan, repair, execute, explain - How planning helps to assemble your home theater. In *Proc. of the 24th Int. Conf. on Automated Planning and Scheduling (ICAPS 2014)*, pages 386–394. AAAI Press.

Susanne Biundo, Pascal Bercher, Thomas Geier, Felix Müller, and Bernd Schattenberg. 2011. Advanced user assistance based on AI planning. *Cognitive Systems Research*, 12(3-4):219–236. Special Issue on Complex Cognition.

Kutluhan Erol, James A. Hendler, and Dana S. Nau. 1994. UMCP: A sound and complete procedure for hierarchical task-network planning. In *Proc. of the 2nd Int. Conf. on Artificial Intelligence Planning Systems (AIPS 1994)*, pages 249–254. AAAI Press.

Juan Fernández-Olivares, Luis A. Castillo, Óscar García-Pérez, and Francisco Palao. 2006. Bringing users and planning technology together. experiences in SIADEX. In *Proc. of the 16th Int. Conf. on Automated Planning and Scheduling (ICAPS 2006)*, pages 11–20. AAAI Press.

Thomas Geier and Pascal Bercher. 2011. On the decidability of htn planning with task insertion. In *Proc. of the 22nd Int. Joint Conf. on Artificial Intelligence (IJCAI 2011)*, pages 1955–1961. AAAI Press.

Alyssa Glass, Deborah L. McGuinness, and Michael Wolverton. 2008. Toward establishing trust in adaptive agents. In *Proc. of the 13th Int. Conf. on Intelligent User Interfaces (IUI 2008)*, pages 227–236. ACM.

Marc Hassenzahl, Michael Burmester, and Franz Koller. 2003. Attrakdiff: Ein fragebogen zur messung wahrgenommener hedonischer und pragmatischer qualitt. In Gerd Szwillus and Jürgen Ziegler, editors, *Mensch & Computer 2003: Interaktion in Bewegung*, pages 187–196, Stuttgart. B. G. Teubner.

Frank Honold, Felix Schüssel, and Michael Weber. 2012. Adaptive probabilistic fission for multimodal systems. In *Proc. of the 24th Australian Computer-Human Interaction Conf.*, OzCHI '12, pages 222–231, New York, NY, USA, November, 26–30. ACM.

Frank Honold, Felix Schussel, Michael Weber, Florian Nothdurft, Gregor Bertrand, and Wolfgang Minker. 2013. Context models for adaptive dialogs and multimodal interaction. In *9th Int. Conf. on Intelligent Environments (IE 2013)*, pages 57–64. IEEE.

Maria Madsen and Shirley Gregor. 2000. Measuring human-computer trust. In *Proc. of the 11 th Australasian Conf. on Information Systems*, pages 6–8. ISMRC, QUT.

Bonnie M Muir and Neville Moray. 1996. Trust in automation. part ii. experimental studies of trust and human intervention in a process control simulation. *Ergonomics*, 39(3):429–460.

Karen L. Myers, W. Mabry Tyson, Michael J. Wolverton, Peter A. Jarvis, Thomas J. Lee, and Marie desJardins. 2002. PASSAT: A user-centric planning framework. In *Proc. of the 3rd Int. NASA Workshop on Planning and Scheduling for Space*, pages 1–10.

Karen L. Myers, Peter A. Jarvis, W. Mabry Tyson, and Michael J. Wolverton. 2003. A mixed-initiative framework for robust plan sketching. In *Proc. of the 13th Int. Conf. on Automated Planning and Scheduling (ICAPS 2003)*, pages 256–266. AAAI Press.

D. Nau, T.-C. Au, O. Ilghami, U. Kuter, D. Wu, F. Yaman, H. Munoz-Avila, and J.W. Murdock. 2005. Applications of shop and shop2. *Intelligent Systems, IEEE*, 20(2):34–41, March.

Florian Nothdurft, Felix Richter, and Wolfgang Minker. 2014. Probabilistic human-computer trust handling. In *Proc. of the Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 51–59. ACL.

Marvin Schiller and Birte Glimm. 2013. Towards explicative inference for OWL. In *Proc. of the Int. Description Logic Workshop*, volume 1014, pages 930–941. CEUR.

Felix Schüssel, Frank Honold, and Michael Weber. 2013. Using the transferable belief model for multimodal input fusion in companion systems. In *Multimodal Pattern Recognition of Social Signals in HCI*, volume 7742 of *LNCS*, pages 100–115. Springer.

Bastian Seegebarth, Felix Müller, Bernd Schattenberg, and Susanne Biundo. 2012. Making hybrid plans more clear to human users – a formal approach for generating sound explanations. In *Proc. of the 22nd Int. Conf. on Automated Planning and Scheduling (ICAPS 2012)*, pages 225–233. AAAI Press.

Shirin Sohrabi, Jorge Baier, and Sheila A. McIlraith. 2009. HTN planning with preferences. In *21st Int. Joint Conf. on Artificial Intelligence (IJCAI 2009)*, pages 1790–1797. AAAI Press.

W3C OWL Working Group. 2009. *OWL 2 Web Ontology Language: Document Overview*. Available at `http://www.w3.org/TR/owl2-overview/`.

8 Appendix A: Example Planning and Dialog Snippet

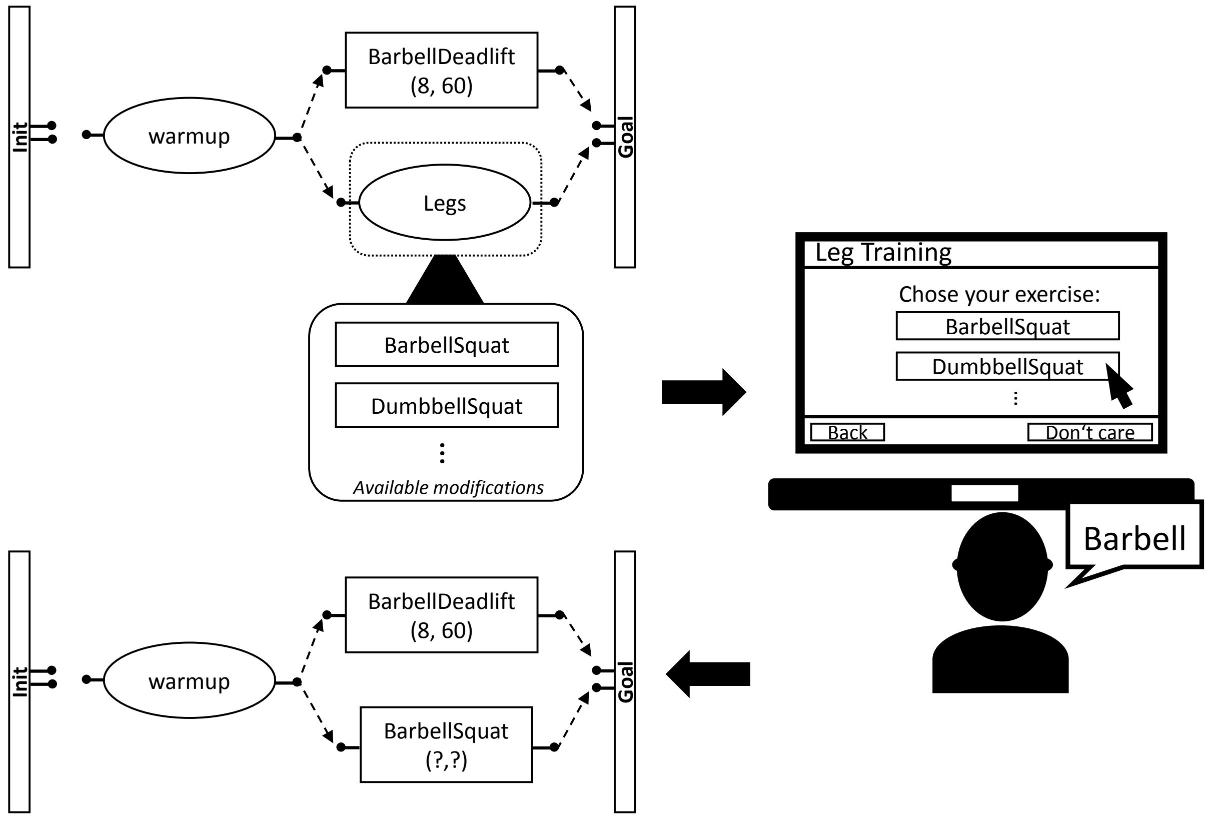


Figure 4: This figure shows an example planning and dialog snippet. The abstract planning task "Legs" has to be decomposed into a primitive task. In this case the decision that of the modifications is to be included in the plan is done by the user. A screenshot of an exemplary decision-making by the user was presented in Figure 2. Afterwards, the abstract task is refined using the selected modification and integrated with the plan.