Knowl Inf Syst
DOI 10.1007/s10115-016-1011-3

REGULAR PAPER

Evaluating intelligent knowledge systems: experiences with a user-adaptive assistant agent

Pauline M. Berry[1,3] · **Thierry Donneau-Golencer**[1,4] · **Khang Duong**[1] · **Melinda Gervasio**[1] · **Bart Peintner**[1,5] · **Neil Yorke-Smith**[1,2]

Received: 15 July 2016 / Revised: 5 October 2016 / Accepted: 18 November 2016

Abstract This article examines experiences in evaluating a user-adaptive personal assistant agent designed to assist a busy knowledge worker in time management. We examine the managerial and technical challenges of designing adequate evaluation and the tension of collecting adequate data without a fully functional, deployed system. The CALO project was a seminal multi-institution effort to develop a personalized cognitive assistant. It included a significant attempt to rigorously quantify learning capability, which this article discusses for the first time, and ultimately the project led to multiple spin-outs including *Siri*. Retrospection on negative and positive experiences over the 6 years of the project underscores best practice in evaluating user-adaptive systems. Lessons for knowledge system evaluation include: the interests of multiple stakeholders, early consideration of evaluation and deployment, layered evaluation at system and component levels, characteristics of technology and domains that

✉ Neil Yorke-Smith
nysmith@aub.edu.lb

Pauline M. Berry
Pauline.Berry@sri.com

Thierry Donneau-Golencer
Thierry.Donneau-Golencer@sri.com

Khang Duong
Khang.Duong@sri.com

Melinda Gervasio
Melinda.Gervasio@sri.com

Bart Peintner
Bart.Peintner@sri.com

1 SRI International, 333 Ravenswood Ave., Menlo Park, CA 94025, USA

2 American University of Beirut, Riad El-Solh, Beirut 1107 2020, Lebanon

3 Present Address: xAd, Mountain View, CA, USA

4 Present Address: Salesforce.com, San Francisco, CA, USA

5 Present Address: Loop AI Labs, San Francisco, CA, USA

Published online: 07 December 2016

determine the appropriateness of controlled evaluations, implications of 'in-the-wild' versus variations of 'in-the-lab' evaluation, and the effect of technology-enabled functionality and its impact upon existing tools and work practices. In the conclusion, we discuss—through the lessons illustrated from this case study of intelligent knowledge system evaluation—how development and infusion of innovative technology must be supported by adequate evaluation of its efficacy.

Keywords Technology evaluation · Knowledge-based systems · Personal assistant agent · User-adaptive · Time management · CALO project

1 Introduction

Not long after the turn of millennium, Bill Gates remarked, "If you invent a breakthrough in artificial intelligence, so machines can learn, that is worth 10 Microsofts" (New York Times, March 2004). Today, it is widely agreed that artificial intelligence (AI) technology has much to contribute to personal convenience and corporate productivity. Learning and adaptation offer prospects for new functionality, together with increased personalization and improved ease of use.

Even so, development and infusion of any technology—not least disruptive technology like machine learning—must be supported by adequate evaluation of its efficacy. Without adequate evaluation, reassessment, and redesign, the knowledge systems deployed risk hindering rather than aiding users, and harming rather than augmenting businesses [41]. For instance, users will likely reject an adaptive system that behaves erratically and unpredictably. Designing and conducting suitable and adequate evaluation poses key challenges, particularly for learning and personalized systems [34,76]. We will use the term 'user-adaptive' for such knowledge systems which personalize themselves to their users [15,53,62], i.e., which build a model of the user and adjust system behaviour based on that model.

In this case study article, we examine experiences in evaluating the learning performance of a user-adaptive personal assistant agent. We present the managerial, design, and execution lessons learned from this case study of evaluation in the light of the literature. The domain is personal time management: in particular, the problem of providing assistance with arranging meetings and managing an individual's calendar. The article primarily concerns the *evaluation* of an intelligent knowledge system that learns preferences over an extended period. As such, the system itself and its performance are secondary topics. Indeed, the evaluation found mixed results, and further we argue that the evaluation methodology was exposed to flaws. Even negative results, however, produce lessons that can aid the evaluation of knowledge systems [24,68]. These lessons as an illustration of the "do's" and "don'ts" of evaluation are our primary contribution.

The technology in our *Personalized Time Management* (PTIME) calendaring assistant agent was designed not only to help with scheduling meetings, but also to help avoid protracted inter-personal negotiations, simplify the reservation of resources, and advocate for the user's time needs in semi-automated negotiation. The agent's usefulness increased as its knowledge about the user increases. The enabling technologies involved were preference modelling and machine learning to capture user preferences, natural language understanding to facilitate elicitation of constraints, and constraint-based reasoning to generate candidate schedules. Human–computer interaction (HCI) and interface design played central roles.

The PTIME system was part of a larger, seminal project, *Cognitive Assistant that Learns and Organizes* (CALO), aimed at exploring learning in a personalized cognitive assistant [58].

Thus, the primary assessment of PTIME was in terms of its adaptive capabilities, although such a knowledge system must necessarily have a certain level of functionality to assist with tasks in time management, in order to provide a context for learning. At the commencement of the project, however, the degree of robustness and usability required to support evaluation was not immediately obvious. Evaluation was focused almost exclusively on the technology: experiments were designed to measure performance improvements due to learning within a controlled test environment intended to simulate a period of real-life use—rather than in a genuinely 'in-the-wild' environment (which we discuss in the sequel). Technologists such as the majority of the authors are trained primarily to conduct such 'in-the-lab' evaluations, but—as we argue in this article—many situations require placing the technology into actual use with real users in a business or personal environment, in order to provide a meaningful assessment. We suggest in particular that the evaluation methodology of CALO gave scant attention to the usefulness and usability of the technology. Indeed, the design and development of the system neglected to involve the user population from early stages: for instance, paper prototypes and Wizard of Oz studies were omitted.

The six lessons that emerged from our evaluation journey with PTIME are not unfamiliar from other experiences of evaluating (non-adaptive) systems:

1. The contexts of the use of technology, and the competing interests of the stakeholders, must be a primary focus in designing an evaluation strategy
2. Evaluating one component based on an evaluation of a whole system can be misleading, and vice versa
3. User-adaptive systems require distinct evaluation strategies
4. In-the-wild evaluation is necessary when factors affecting user behaviour cannot be replicated in a controlled environment
5. In-the-wild evaluation implies significant additional development costs
6. Ease of adoption of the system by users will determine the success or failure of a deployed evaluation strategy

We illustrate these lessons through the article and conclude with recommendations for technology projects and their evaluation. Our take-home lessons are generic and are not specific to the CALO project. In fact, one of the main lessons from our case study of intelligent knowledge system evaluation is that researchers and project managers can be insufficiently familiar with evaluation methodologies and best practice, and that evaluation is too often included towards the end of a project rather than made an integral part of the development cycle.

We begin by introducing the problem domain of time management (Sect. 2) and summarize the final design and capability of the PTIME system, highlighting the potential value of AI technology in the domain (Sect. 3). Section 4 reports and critiques the design, execution, and outcome of the *evaluation* of PTIME, relating our experiences in attempting to assess the system's learning capability over six calendar years of the CALO project. We conclude the article with an analysis of our experiences in the light of the literature and other evaluation efforts (Sect. 5), and with the lessons illustrated by the PTIME evaluation case study (Sect. 6).

2 Domain: personalized meeting scheduling

The vision guiding the CALO project was a personal assistant for a busy decision-maker in an office environment, akin to an adept administrative assistant who can help facilitate routine tasks [58,72]. CALO thus fits into the heritage of user-assistive agents [5,13,19,25,30,43,53,

55,60,70,80,82,83]. From a scientific standpoint, a primary objective was to stimulate the development of AI learning technology to support autonomous adaptation in such assistive agents.

The CALO project and follow-on technology transitions spanned the period 2003–2010. With hindsight, CALO has been judged a success [1,12]. The project stimulated research into machine learning, established the concept of a personal assistant agent into the mainstream, and led to now-standard toolkits and datasets (e.g., [46]); and from it came commercial spin-outs as well as knowledge systems adopted by several branches of the US government [28,65].

Indeed, while CALO was primarily an academic research effort, several spin-out companies developed, including *Siri*, which was subsequently acquired by Apple [12], and smart calendar app *Tempo*, which was recently acquired by Salesforce.com [48]. The capabilities of Tempo centred on a calendar display interface, the use of machine learning to associate documents, and semantic entity recognition and the use of web services to, for instance, find driving directions for a meeting.

In the context of the vision of CALO, time management—in particular, scheduling meetings in overconstrained settings which are difficult for people to resolve—was a natural problem to address. The domain of time management is crucial for CALO's target user population of busy knowledge workers. The reality is that scheduling office meetings too often turns into a tedious process. Protracted negotiations occur as potential participants communicate in the semi-transparent world of their own and others' calendars. The industry that exists around time management and calendaring—tools (both paper- and computer-based), strategies, and books—is predicated on the valuable time that could be saved if the effort spent organizing meetings could be reduced [36,55]. In a world of electronic calendars and advancing AI technology, the prospect of automated or semi-automated scheduling assistance seemed a plausible and desirable application of a knowledge system; yet in the early twenty-first century, the most common practice for negotiating meetings remained extended email, phone, or instant messaging interactions.

Both commercial (e.g., Microsoft Outlook) and open-source (e.g., Zimbra) calendaring systems abound to support centralized calendaring solutions within an institution. However, these tools leave to the user the task of identifying meeting options and choosing among them, and negotiating with invitees. At best, more advanced tools support the user in selecting a time by graphically depicting the availability of participants. Such group calendaring systems are strong in integration with institutional workflow and enterprise systems , but they remain tools rather than providers of intelligent assistance.

Little has changed since PTIME was conceived. The nuggets of AI technology that have been finding their way into mainstream calendaring tools—such as the entity extraction and limited natural language parsing in Google Calendar—underline the potential of AI for the time management domain. However, intelligent, personalized time management assistance is difficult: user studies indicate that people seldom have a realistic understanding of the preferences they exhibit in practice [79], few are willing to keep providing explicit feedback without seeing tangible benefit relatively quickly [18], and most are unwilling to spend time training a calendaring system [55,81]. As a result, we designed PTIME to be an adaptive system that would learn its user's preferences reasonably quickly and non-intrusively through implicit feedback.

From a technological point of view, there already existed fully or semi-automated scheduling systems that readily solved the multi-participant, distributed or centralized meeting scheduling problem (see the discussion in Berry et al. [10]). However, they continue to suffer from low adoption rates because they fail to account for the intensely personal nature

of scheduling, or because they demand excessive control of an important aspect of the individual's life [63].

Systems subsequent to PTIME have been developed in the context of ubiquitous mobile computing, addressing, for instance, issues of complex multi-event scheduling [66], privacy of ambient calendaring [69], or multi-agent negotiation strategy [70].

As a research project, the objective of PTIME was to develop an intelligent personalized calendaring assistant for busy knowledge workers. The broader objective that gained emphasis as the project progressed (and as the technical challenges were addressed) was to develop a usable personalized assistant that would be useful in helping to ameliorate time management problems. Providing intelligent time management assistance required that we either augment an existing calendaring system or develop a fully functional, stand-alone assistant. We initially attempted to use Microsoft Outlook as our calendaring interface—encountering and overcoming significant integration issues—but eventually set aside that paradigm when CALO began to develop its own calendaring interface [20], in line with the vision of CALO as a separate, cognitive knowledge system.

3 Evaluation target: the PTIME system

PTIME consists of four main components: user interface, calendar proxy, constraint reasoner, and preference learner. Through PTIME's *user interface* (UI), shown in Fig. 1, the user browses and manipulates her calendar, and enters scheduling constraints and meeting details using any combination of typed restricted natural language and direct manipulation. The *calendar proxy* provides the ability to connect to a variety of calendar servers, supporting PTIME's ability to manage calendars from multiple sources (e.g., personal and work calendars). The *constraint reasoner* generates candidate schedules using the current prefer-

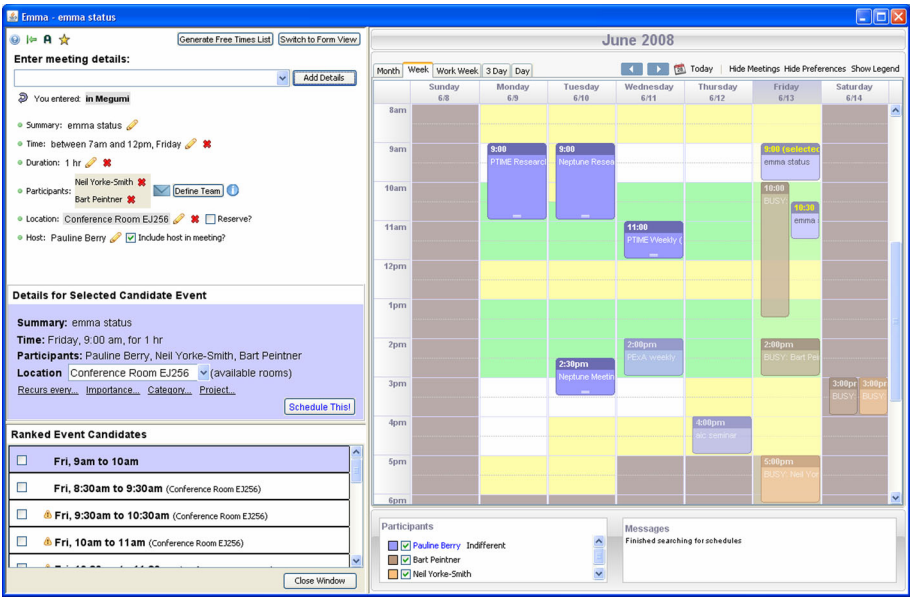

Fig. 1 One view of the PTIME interface near the end of the project (PTIME-5)

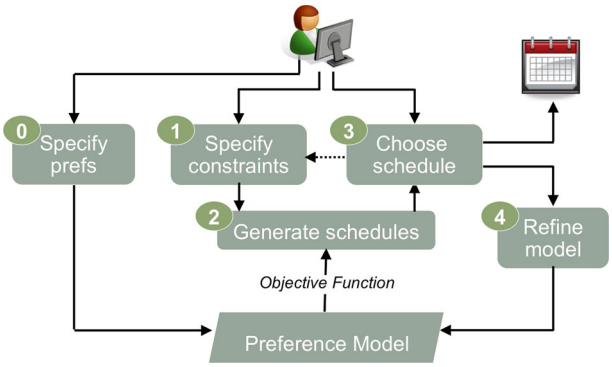

Fig. 2 Overview of user–system interaction in the PTIME adaptive time management assistant. The primary interaction is (*1*) user's specification of desiderata for a meeting, (*2*) system's generation of preferred schedule options, and (*3*) user's choice among the options or refinement of desiderata. The system (*4*) updates its preference model based on the chosen option

ence model while the *preference learner* updates this model based on user feedback on the generated options. Berry et al. [7,10] provide a detailed system description.

Figure 2 depicts the interactions between the user and the PTIME system during a typical scheduling process. Optionally, PTIME (Step 0) may elicit scheduling preferences: for the user, this corresponds to specifying general preferences such as preferred days and times. The elicitation methodology and 'wizard' user interface are described in Berry et al. [10]. From the system's point of view, the purpose of this optional user step is to initialize the preference model. The form of the model is described below. At any time the user can reinvoke the preference wizard. In order to avoid confusion between stated and learned preference models, the system displays the stated model, i.e., the user's stated general preferences. The user can choose to modify them, and the system takes this change into account.

In its main mode of operation, PTIME (Step 1) elicits an event request: for the user, this corresponds to stating details on the desired event to be arranged; these details correspond to constraints. This can be seen in the upper-left part of Fig. 1. PTIME handles single calendar 'events', such as a meeting, and recurrent editions of single events. The system is not designed to handle events with multiple parts (compare SelfPlanner [66]) nor plan daily agendas.

PTIME (Step 2) computes preferred candidate schedules (possibly relaxations) in response to the request and presents a ranked subset of the candidate schedules to the user. These can be seen in the lower-left part of Fig. 1. Note that, because PTIME will consider moving existing events if necessary, the options presented to the user are schedules rather than single events. The number of such candidate schedules presented depends on the number of feasible schedules. PTIME will typically display 10 candidate schedules, including a mix of highly optimal and diverse options. PTIME (Step 3) accepts the user's selection from among the presented candidate schedules. This is seen in the centre-left part of Fig. 1, where the user will click "Schedule This!" if satisfied with the selected candidate schedule. The candidate schedule is also previewed in the right-hand calendar view as semi-transparent events. These become firm once the user selects the candidate schedule. PTIME (Step 4) updates the preference model accordingly, as described below.

Steps 2 and 3 repeat as necessary, with the system presenting new or refined options after each new detail is entered by the user. The updated model is used in the subsequent interactions. Through a collaborative negotiation process (not depicted), event invitees comment,

respond, and counter-propose to reach agreement over the event [9]; they can delegate some of this negotiation to their PTIME agent to perform on their behalf as a mediating agent [70].

Underlying the constraint reasoner and the preference learner is a *preference model* designed to be expressive enough to capture the user's preferences while being simple enough to support tractable reasoning and efficient learning. The preference model is a second-order Choquet integral model over event request and schedule features. This a well-known model for making decisions under multiple criteria [32]; the PTIME preference model has 28 real-valued coefficients. At any time, the model is a weighed combination of the initial stated model (or an uninformative default if the user does not state general preferences) and the acquired model.

The *constraint reasoner* generates scheduling options in response to new or revised details and constraints from the user, using the current preference model to generate preferred options. The reasoner translates requests such as *"next tues afternoon with nigel and kim"* into a set of soft constraints and solves a soft constraint problem with preferences [10,57]. Soft constraints allow all aspects of the user's request—including times, location, and participants—to be relaxed in the case where the request cannot be satisfied, i.e., when the scheduling problem is overconstrained. For details of the constraint solving, we refer to the cited works.

The *preference learner* interacts with both the UI and the constraint reasoner. In addressing an event request from the PTIME user, the reasoner queries the learner for the current preference model, and uses the model to generate candidate solutions. After the user selects an option from those presented, the UI sends the candidate set to the learner, providing it with the feedback that the selected option is preferred to all the other displayed candidates. Although users have the ability to provide explicit ratings on any of the candidates, the learner is designed primarily to learn unobtrusively from implicit feedback, i.e., user decisions and actions. Given these pairwise preferences, the learner applies SVM techniques [42] for ranking to adjust the weights of the evaluation function used by the constraint reasoner (see Fig. 2). Additional learning components acquire models of autonomy, preferred locations, social relationships, and event reminders.

4 Methodology: evaluating learning in PTIME

We now describe the methodological approach to evaluating the PTIME system and critique the various specific evaluations undertaken. The lessons stated in Sect. 1 emerge; we discuss them in detail at the end of the article. It is important to keep in mind that our focus is on the lessons illustrated by a case study in evaluation: the results of the specific evaluations themselves are secondary. The reader interested in the take-home lessons from our experience can skim the remainder of this section and move to Sects. 5 and 6.

Different aspects of PTIME were under development over 6 years as part of the CALO project [72]. During that time, we undertook a number of evaluations of the technology focused on its adaptive capabilities, since learning was the major thrust of the CALO project. For various reasons which we discuss next, the evaluation turned out to be a much more demanding task than originally anticipated, requiring a rethinking not only of our evaluation methodology but also of the design of PTIME itself and the scheduling assistance capabilities it provided.

For clarity given the scale of the project, we append the project year number to the end of 'PTIME' or 'CALO' to indicate the respective system versions at the end of the project

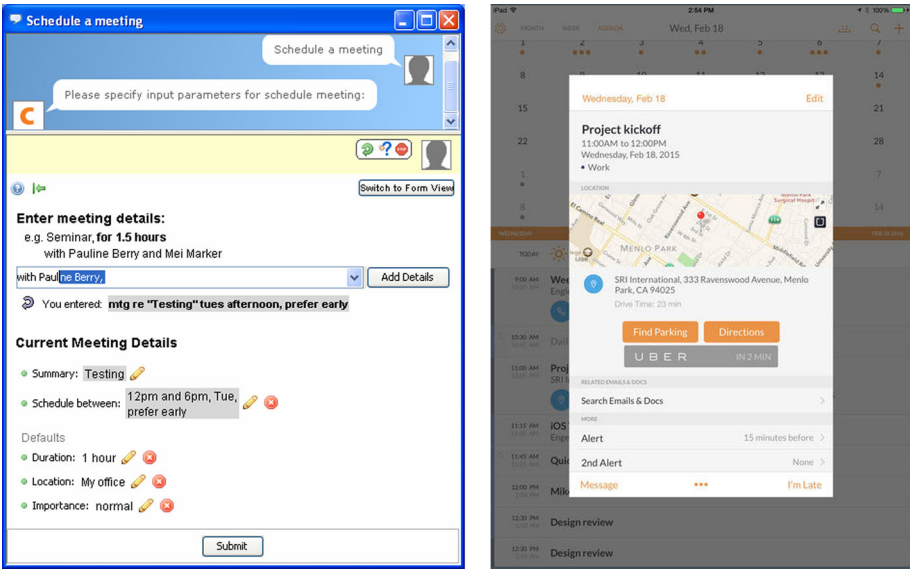

Fig. 3 *Left* Main PTIME interface, end of Year 3 (PTIME-3). *Right* PTIME spin-out *Tempo*

'Year'. (For administrative reasons, each of the five 'Years' of the project were approximately 14 months in duration.) For example, PTIME-2 was the version of PTIME within CALO-2 at the end of project Year 2, the system that took part in the first annual CALO Test, described below.

Our experience was inherently within the context of the CALO project. For the first half of the project, we relied primarily on the CALO-wide evaluation. Section 4.1 critiques this 'Years 2 and 3' evaluation. (Year 1 did not have formal evaluation.) With hindsight, dedicated requirements elicitation for and informal evaluation of PTIME would have been beneficial (Lesson 2). They would have helped set the direction of PTIME's development, such as a calendar-based user interface. The first two years of effort on PTIME, however, concentrated solely on algorithmic research and infrastructure building, and the first CALO-wide evaluation occurred towards the end of Year 2 of the project; further work led to PTIME-3, whose user interface is shown in Fig. 3 (left). During the subsequent project Year 4, in addition to the CALO-wide evaluation we undertook a separate informal evaluation of PTIME, and made a limited deployment of the system for evaluation purposes. Section 4.2 critiques this evaluation. Finally, in the last Year of the project, we were free to develop and evaluate PTIME on its own. Section 4.3 critiques this evaluation.

Relevant to but outside of our prerogative as researchers developing PTIME were matters of CALO project management, including the scope of the overarching project and its research objectives, and the evolution of the project objectives and requirements over time (Lesson 1). Also outside our prerogative was the purpose, design, and conduct of CALO-wide evaluations (Lessons 5 and 6). We describe them below from the perspective of PTIME.

4.1 Years 2 and 3: evaluating PTIME as part of CALO

As stated, a primary objective of the CALO project was to stimulate the development of learning technology to support adaptation in an intelligent assistant. Therefore, the annual *CALO*

Test, inaugurated in Year 2 of the project, was designed to assess the effects of learning on CALO's performance as a personalized office assistant. The methodology of the CALO Test was—for better or worse—dedicated to make this assessment. Patterned after standardized tests such as the SAT and GMAT for American college admissions, each year's Test drew from a library of parametrized questions developed by an independent evaluator (IE).

We give an abbreviated overview of the Year 2 and 3 CALO Tests in order to understand its relevance to the evaluation of PTIME (all Lessons 1–6). The methodological approach of the CALO Test was a kind of layered evaluation [14,64]. The Test was significant as an attempt to rigorously quantify the learning ability of a personal assistant agent, in the context of a large, multi-institution effort to build a groundbreaking cognitive knowledge system. Since the evaluations were similar for Years 2 and 3, we describe them together.

Aims. The aim of the CALO Test was to assess the effects of learning on CALO's performance as a personalized office assistant, and hence PTIME's performance as a personalized time management assistant. It is important to note that *task performance as a result of machine learning* was the only criterion of interest. In particular, the CALO Test did not consider usefulness, user opinion, or usability, other than their implicit impact upon 'performance' as defined by score on the Test questions (see below). Indeed, if the aim had been to evaluate the usefulness of the technology, the design of the Test would have had to be different; we return to this point later. For the perspective of PTIME, the Test was mandated; we had no influence over its design (Lessons 1 and 2).

Subjects. In order to provide data for the machine learning algorithms, subjects used CALO in their day-to-day activities during a defined period, called the *critical learning period* (CLP). Participants were recruited from various departments within our organization. All were knowledge workers; most had some role in the CALO project. In Year 2, 15 participants were recruited; in Year 3, 16 participants.

Hypotheses. Three main hypotheses were studied:

1. CALO with learning enabled (*LCALO*) has a higher performance score (defined below) than the baseline of CALO with learning ablated (*BCALO*).
2. The scores of both LCALO and BCALO increase over the years of the project.
3. The difference in scores between LCALO and BCALO increases over the years.

Note that BCALO as well as LCALO were developed over the years, i.e., BCALO was not a standing comparison target for LCALO but should itself be improved.

Methods and tasks. We refer to "Appendix 1" for the detailed description.

Results. We report the results for PTIME in Table 1, which shows the mean percentage score (i.e., the mean score divided by the maximum of 4.0: e.g., a mean score of 2 is 50%). Columns 'Target' denote the percentage score required for continuation of the project into the next

Table 1 Overall PTIME-2 and PTIME-3 score results for the CALO Test in Years 2 and 3

System	Year 2		Year 3	
	Target (%)	PTIME PQs (%)	Target (%)	PTIME PQs (%)
LCALO	32	65	46	69.5
BCALO	22	62	26	65

project year; columns 'PTIME PQs' denote the actual percentage score achieved by PTIME on the CALO Test *parameterized questions* (PQs) relevant to time management. PTIME's performance was well above the target levels. The differences between LCALO and BCALO between each user, and overall averaged across users, although small in magnitude, were both statistically significant at $p < 0.05$ (two-sided t test). For CALO overall, the performance of LCALO over BCALO was positive but not statistically significant.

Discussion of the years 2/3 CALO Test The CALO Test represented a hybrid between a theoretical assessments of machine learning in a cognitive assistant [71] and assessments based on human subject experiments [74]. Despite various critiques and inadequacies, the CALO Test was mandated by the project sponsor. First and foremost, the CALO Test was designed specifically to evaluate CALO as a whole rather than any of the learning components in isolation. Given the large collection of modules, each year's instance of the Test could include only a small number of questions pertaining to time management.

Second, the data gathered during the critical learning period was insufficient for PTIME's learning. In the first year of the annual evaluation (Year 2 of the project), participants simply did not schedule enough events. In Year 3, participants were directed to schedule at least a minimum number of meetings, but they ended up scheduling mostly underconstrained meetings (a common type of meeting request in practice, as we discuss below). As a result, PTIME-2/PTIME-3 did not receive data on how they managed trade-offs, which is most indicative of user preferences. Finally, usability and stability issues—with PTIME and with CALO overall—significantly affected user interaction, impacting our ability to gather data (Lesson 6).

We undertook an analysis of PTIME's performance each year, to investigate the mistakes made by the system on PQs and their causes. Generally, apart from some bugs, the causes were attributable to incorrect or incomplete preference elicitation, inadequate data for learning, or the nature of the Test as we next discuss.

The constraints of a mandated evaluation process did not provide us with control over our own evaluation regime. This said, although PTIME was mandated to participate in the CALO Test, there was nothing in the mandate that explicitly prevented us from undertaking our own evaluation—and indeed later we began to do so, as described below. In hindsight, at the time the Test was first conceived and before details about the Test and its execution were laid out, we imagined that it would be a way to have controlled access to a reasonably large group of people using our system, and thus a good forum for evaluation (Lessons 3 and 4). Further, it soon became clear that running such a large-scale evaluation on a complex, adaptive system held considerable challenge, and so, in some sense, it seemed an even better idea to pool resources so that all could benefit from the large-scale evaluation. Indeed, we would not have had the resources to run a dedicated PTIME evaluation at the time, even if we had thought to do so (Lesson 5).

The advantage of evaluating PTIME as a component in a larger system, then, was access to a large-scale independent evaluation. Further, since CALO's ambition was to be a personalized office assistant, connectivity and synergy between different components was thought important to give assistance in a broad range of tasks. For example, CALO learned organizational hierarchy and this information was available to PTIME to help answer questions about event attendee importance. The disadvantage of evaluating PTIME in this regime, as we next discuss, was the difficulty of assessing the performance of PTIME as opposed to the performance of the larger system (Lesson 2). This difficulty led us to eventually develop our own evaluation (see Sect. 4.3).

"Appendix 2" summarizes further critique of the CALO Test methodology. Between Years 2 and 3, as we began to realize the difficulties with the CALO Test from the point of view of PTIME, and to recognize the increasing importance of usability, we conducted our first user study to investigate calendaring needs and relevant decision factors in the target user population of knowledge workers.

Take-home summary: Generic system-level evaluation may be unsuitable, even if scientifically conducted.

4.2 Year 4: evaluating PTIME, besides the CALO Test

Summarizing, the attempt to evaluate PTIME as a component within CALO, by means of the annual project-wide evaluation in the first years of CALO—this proved to be an unsuccessful evaluation strategy. Lessons 1 and 2, regarding context, stakeholders, and local versus global evaluation, became prominent.

In project Year 4, we sought to address deficiencies of the earlier evaluations (i.e., the CALO Test) from the point of view of PTIME, by a three-pronged strategy: a pre-CLP training phase (Sect. 4.2), post-CLP interviews (Sect. 4.2), and a limited deployment of PTIME-4 (Sect. 4.2). Our aims were to obtain a greater amount of data to assess PTIME-4's learning performance, and to gain insight into PTIME-4's efficacy and usability. The main system developments in PTIME-4 were increased efficiency in the constraint reasoner, and improvements to the user interface as a response to deficiencies noted in Year 3 Test.

With hindsight, it is surprising that a large-scale research project like CALO did not feature usability more prominently in its initial project plan, and plan for user-centric evaluations as soon as the system had reached a minimal adequate level of capability (both task functionality and learning capability) and user interface. To be fair, the strong mandate for the annual evaluation was on task performance as a result of machine learning. The overall project plan could be characterized by an over-simplification as: first develop the learning technologies and then integrate them and test and iterate.

With others, we put forward the case that became apparent from the Year 2 and especially Year 3 CLPs: that data collection and task performance were both hindered if the CALO system suffered from poor usability. In tandem, the project requirements did evolve in Year 4 and especially Year 5 of CALO , onto a focus towards deployment of parts of the system or of concepts from it (Lesson 1).

Pre-CLP training and CALO Test results In Year 4, first, we attempted to make the best we could of CALO Test, introducing additional training prior to the CLP.

Table 2 shows the CALO Test results. The performance of LCALO-4 over BCALO-4 was, as in previous years, positive but small. We determined two main reasons why LCALO performs only a little better than BCALO. The first is that there was little time during the CLP for the effect of learning to be seen. The second reason is that the baseline performance of BCALO was already high on the PTIME PQs. The main reason for this is the selection of the PQs (and their scoring—see Sect. 4.1): many of the PTIME PQs were relatively easy

Table 2 Overall PTIME-4 score results for the CALO Test in Year 4

System	Year 4 target (%)	PTIME PQs (%)
LCALO	64	73
BCALO	32	71.5

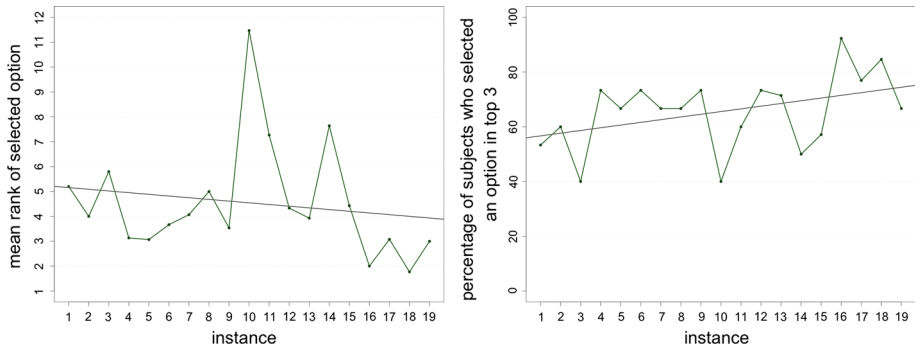

Fig. 4 Results of the year 4 evaluation. *Left* Options earlier in the system's ranking should be selected by users as the system learns. *Right* Users should be satisfied with the first-presented option more frequently as the system learns

for a non-learning algorithm to score well on. For the reasons discussed earlier, despite the addition of five more PQs, the CALO Test did not include many other PQs which were arguably more difficult and where learned knowledge would weigh more heavily.

Our changes to the experimental setup were successful in that a significant portion of the data collected now involved users making trade-offs in overconstrained scenarios. Figure 4 illustrates part of the results. The left graph shows the learning curve based on the rank of the selected option, averaged across the 15 completed subjects. The results suggest a gradual improvement in the preference learner's ability to rank highly (i.e., lower absolute rank) the most desirable option. We also measured the number of times the subjects' selected option was in the top 3 options presented (right graph, averaged across the 15 subjects). The results suggest that, with more learning, the selected option occurs in the top 3 more frequently. Combined, the results given an indication that PTIME-4 was able to generate and present reasonable options to the user.

In terms of learning, while the results were encouraging, they are not dramatic. The learned models arguably still did not truly reflect the participants' scheduling preferences (as stated in post-CLP answer key elicitation), again for a number of reasons. First, in the dedicated training phase, although the details of each scheduling scenario were spelled out, it remained up to the participants to internalize the details and pretend as if they had created that scenario. In reality, CALO Test participants were often forced to make trade-offs in scenarios they would have never created. Second, in the CLP, the fact that the participants knew they would not have to actually *attend* most of the meetings they were scheduling sometimes had a dramatic influence on their perception of what was acceptable. Third, the timescale for adaptivity was arguably too small.

Post-CLP interviews Following the CALO Test, second, we interviewed the participants and others. Participants who had used PTIME (any version) felt that the concept of a personalized calendaring assistant along the lines of PTIME was useful (mean 1.0 on a scale $[-2, 2]$, significant by a one-sided t test against indifference (mean 0) at $p < 0.001$), and found that they could express their meeting requests using it (mean 1.0). Opinion about the user interface was much more mixed (mean 0.15, not significant at $p < 0.1$). Participants also did not have a sense of whether or not PTIME-4 was adapting to their preferences (mean 0.33)—see discussion below—nor did they express significant opinion whether or not PTIME could fit into their preferred time management workflow (mean 0.35). By contrast, subjects expressed

that the organization's enterprise calendaring system did not fit into preferred workflow (mean -0.69, $p < 0.05$).

The interviews indicated that participants desired to understand why PTIME-4 made its recommendations, and in particular why the system sometimes relaxed constraints that they had specified. Subjects liked the PTIME-4's visualization of their preferences. However, they were often unaware when their meeting requests were overconstrained. This found desire for transparency supports studies reported in the literature [26,29]. We modified PTIME to make more prominent alerts about constraint relaxation (see Fig. 1), and to provide mouse-over explanation.

Internal deployment of PTIME Finally, we attempted a limited deployment of PTIME-4 within our department, under the supposition that if any part of CALO would be used in real life , PTIME would be a start. The rationale was to gather data from users in a more 'in-the-wild' evaluation compared to the semi-artificial conditions of the CLP (Lesson 4). Such a deployment required separating PTIME-4 from some of the heavyweight core of CALO-4 and connecting PTIME-4 with our organization's enterprise calendaring system (Lessons 1 and 5); thus we began our recognition of usability as a more significant factor in evaluation as a system moves towards deployment.

The limited deployment of PTIME-4 was almost entirely unsuccessful for three reasons (Lesson 6). First, although much improved over Year 3, the UI, coupled with CALO-4 system performance and stability issues, continued to hamper usability. Second, the installation was still quite heavyweight, requiring some memory-intensive CALO-4 components, such as the knowledge base [3]. Third, while PTIME-4 provided advanced scheduling features, it did not provide sufficient advantage for users' simpler, more common calendaring needs such as inserting a fixed-time appointment—the 'easy' cases [59]—to entice people to use it in conjunction with their existing tools and calendaring habits (Lesson 1).

Overall, the results from the various Year 4 evaluations reflected tension between research objectives and system requirements (Lesson 1). For example, from a research point of view, 'hard' scheduling problems hold greater interest and challenge than 'easy' cases. However, 'easy' cases are common in real life, and users sought a capability for them—hence an assistant must be adept in both situations.

Regarding the assessment of PTIME-4's adaptation to its user, the Year 4 evaluations provided some limited evidence. Although CLP participants overall had only a weak perception that the system was adapting to their preferences, a sizable minority strongly felt that it was doing so. Quantitative data from the Test gave indication of learning. Performance on Test questions was encouraging, within the confines of the Test questions, scoring scheme, and timescale.

The evaluations gave further data on time management practices and calendaring tool usage. Regarding the potential usefulness of PTIME-4, they indicated potential. Regarding usability and deployment, the evaluations confirmed that engineering and user interface work was needed on PTIME-4 (Lessons 5 and 6). These points lead into Year 5.

Take-home summary: Evaluation in controlled settings may be inadquate to address all key evaluation criteria.

4.3 Project year 5: comprehensive evaluation of standalone PTIME

Summarizing, the evaluations conducted in Years 2/3 and 4 of the CALO project gave some insights into PTIME's performance and value, but yielded insufficient data to evaluate the usefulness of PTIME's adaptive capabilities. In Year 5, the annual CALO Test was not

conducted, due to an emphasis on technology transition at the end of the project [17,65]. This enabled us to step back and create an evaluation plan specifically for PTIME-5 (Lesson 2). Learning from our earlier experiences, we designed a layered three-stage evaluation with the objective of answering these research questions:

1. **Is PTIME a usable calendaring assistant?** (user experience)
2. **Does PTIME increases task effectiveness?** (objective performance measures, including measures of learning)
3. **Is PTIME a potentially useful assistant?** (subjective appraisal and performance measures)

The first stage of the Year 5 evaluation involved cognitive walkthroughs, think-aloud exercises, and prototyping. The second stage comprised of a set of structured evaluation sessions (Sect. 4.3). Together, these studies focused on gathering data to assess usability and perceived usefulness, and helped us evolve the UI design. Deliberately, unlike the CLP, we did not attempt to gather quantitative data to assess preference modelling or learning by means of these structured studies. The third stage was a longitudinal study with a small set of users 'in-the-wild' (Sect. 4.3). This study sought to gather data to assess learning performance as well as long-term perceived usefulness (Lesson 4). This stage of the evaluation was truncated due to the conclusion of the CALO project.

Before PTIME-5 could be evaluated 'in-the-wild', the architectural, deployment, and usability issues that hindered previous evaluations had to be addressed. We undertook an iterative development process alongside the first two stages of evaluation, using the results of the user studies, think-aloud exercises, and prototyping, in order to develop alpha, beta, and release deployments [81].

The result was a lightweight, user-centric version of PTIME called *Emma* (*Event Management Assistant*). Emma operated independently of the full CALO system; it operated cross-platform on Mac OS and Windows; and it interfaced with standard iCalendar or CalDAV servers, as used in most institutions. Although the separation from CALO limited some functionality—for example, not having access to the CALO knowledge base, Emma lacked knowledge of organizational hierarchy—it was clear that adoption of PTIME would be greatly inhibited if deployment required the full CALO system to be installed (Lesson 6). We redesigned the calendar view for usability (Fig. 1) and added support for typical calendar functions to address the 'easy' cases of scheduling. Learning from the think-aloud exercises, we made the user–system interaction more incremental and redesigned the request specification and option viewing interface to make less cumbersome the 'hard' cases. There was significant effort required in designing and developing Emma, not least in redesigning the UI, making Emma a standalone application, and integrating it with standard calendar servers (Lesson 5).

Structured evaluation. Even more than the CALO Test of previous Years, the structured evaluation took place 'in-the-lab' in a controlled environment. The objective of the structured evaluation sessions was to evaluate Emma empirically in terms of ease of use and perceived usefulness, factors crucial for eventual user acceptance.

The majority of participants thought that Emma was a potentially useful tool, helping them to manage their calendar more quickly and with superior outcome. Almost all subjects liked the concept of "a tool like Emma" (mean 0.92 on a scale $[-3, 3]$, significant by a one-sided t test against indifference at $p < 0.01$) and "would use it if it was stable" (mean 2.25, $p < 0.001$). Emma was found to be extremely significant in increasing task effectiveness in general (mean 0.85 on a scale $[-2, 2]$), and in terms of perceived speed (mean 0.76) and quality (mean 0.71). It was likewise found easy to use (mean 0.72). Emma was found

to be neither easy nor hard to learn; likewise with perceived understanding and control of the system. As expected from the short duration of usage, participants did not have a sense of whether the system was adapting to their preferences. This result was expected, not because the system's behaviour was not transparent or because the preference model was not scrutable enough, but because the short duration would not give enough scheduling instances and calendaring usage for PTIME to learn preferences. This was one motivation for the longitudinal study, reported next. Finally, the participants who had used an earlier version of PTIME expressed strong agreement that Emma was "superior to PTIME" (mean 1.33). We attribute this perception to Emma's lightweight nature, compatibility with existing calendar systems, and UI improvements.

While these studies on usability and perceived usefulness were not seeking to address the issue of adaptation, we firmly believe they are critical to the evaluation of an user-adaptive assistant such as PTIME. Indeed, as we discovered in our earlier technology-centric evaluations, it is difficult to synthesize scenarios that will effectively elicit real-world behaviour from users within a controlled setting (Lesson 4). On the other hand, evaluating a system 'in-the-wild' requires that it be usable and useful enough for users to actually want to use it (Lesson 6).

Longitudinal study The structured evaluation served as prerequisites to the longitudinal study that sought to evaluate Emma's adaptive capabilities. The longitudinal study focused on a small set of real users 'in-the-wild' over an extended period of three months. The aim was to gather data to assess learning performance—only possible through extended use, as we have argued—as well as long-term perceived usefulness (Lessons 3 and 4).

The longitudinal study was truncated, unfortunately, due to a redirection of funding towards the conclusion of the CALO project. Further, the quality of the data recorded proved variable, due to a number of experimental and technical factors (Lessons 1, 5, and 6). First, some of the participants did not actually use Emma, for reasons varying from summer vacation to technical problems (e.g., firewall access to enterprise calendar servers in other locations). Second, the work patterns of some participants did not fit the target population of the system, despite what the participants had stated when they were recruited. For example, one group of participants who worked together as a team expressed great approval of being able to consult each others' calendars and check room reservations using Emma. They used the system frequently for this purpose. However, they only occasionally used the system to schedule events (preferring pen-and-paper calendars attached to the door of conference rooms), and thus the system obtained little data on user preferences. Third, technical issues with data logging confounded initial preference elicitation and training data with logged data over time.

Some tentative conclusions can be drawn from the longitudinal data; because we do not wish to place too much weight upon it, we refrain from presenting numerical results here. The logged learning data, after cleaning, shows that the rank of the chosen option or options decreased over time [(compare Fig. 4 (left)], suggesting that the system is successfully adapting to its user's preferences. Moreover, the first-ranked option was frequently chosen by all participants. This indicates that the options presented by the system are satisfactory for the meeting request entered by the user; it also indicates that users can satisfactorily express their requirements through the user interface.

Take-home summary Non-technical factors are as pertinent as technical factors in the success of knowledge system evaluation.

Despite the various limitations, we hold that the experimental methodology towards the end of the project was appropriate, notwithstanding some flaws. The first two stages informed

iterative design of the system, and provided data on user experience and task effectiveness in the lab setting. The third stage provided data on learning performance out of the laboratory setting (Lesson 4) (compare Gena [31]).

Had we the freedom to embed an evaluation methodology in project's plan from the beginning, it would have emphasized an iterative development process where the concept, interface, and system capability (constraint reasoning, preference learning, and schedule presentation and recommendation) co-evolve. Such a 'double-helix' process involving the user population was adopted successfully in the technology transitions coming out of the CALO project [28,65]. The plan would include ethnographic user studies and concept design and validation at the start of the project, and separate evaluation of PTIME in terms of usability, task effectiveness, and perceived usefulness throughout. The longitudinal study would have been piloted and then initiated by Year 4, so that it could run to completion before the end of the project. Managerial decisions would continue to provide enabling resources. Ideally, legal issues would have been resolved early so that the user pool could extend outside of our organization.

5 Discussion: evaluating an adaptive, interactive system

Having critically examined our evaluation experiences with the PTIME system in project Years 2/3, 4, and 5, respectively, in this section we reflect more broadly; in Sect. 6 we conclude with the lessons learned.

PTIME was developed in the context of a non-profit research laboratory. The CALO project of which it was a part was applied academic research , having both research and development as integral parts. Our evaluation experience reported here was influenced by the interplay of these parts—research and development—and between shifting project goals and evaluation criteria (Lesson 1). A similar experience among research projects on a trajectory towards deployment could, we suppose, be told by many researchers from the industrial research laboratories. However, such stories are rarely told, McCorduck and Feigenbaum [54]'s overview of Japan's famous Fifth Generation Project being a notable exception.

As we described, the CALO Test aimed to provide an objective, quantitative empirical measure of the effects of learning on CALO's performance. It adopted the 'scientific' approach articulated in the literature on empirical AI [24], which majors on robust evaluation of algorithms and approaches, primarily to show that they improve performance in some way: increased predictive accuracy, efficiency, precision/recall, optimality, area under the ROC curve, and so forth. However, when the contribution made by the technology is to improve a user's experience or adapt to a human's needs or preferences over time, traditional evaluation metrics, e.g., precision/recall, are less obviously applicable [75]. Davis et al. [27] famously argue that it is more important to evaluate *usefulness*, i.e., the extent to which the technology impacts the usefulness or utility of the application; the CALO Test did not do so, while the final evaluation of Emma did strive to do so.

Taking a broad view of evaluating adaptation an intelligent knowledge system poses several key questions. We structure this section around them, before remarking on three related areas of evaluation case studies.

Control conditions for evaluation. First, the appropriate control conditions for proper evaluation. For an adaptive system, the most relevant conditions might be: (1) the adaptive system (e.g., LCALO), (2) the same system but not adaptive (e.g., BCALO), and (3) a 'business as usual' case (e.g., Outlook).

It is illustrative to consider our case study here. The CALO Test rigorously compared conditions (1) and (2) but did not consider condition (3). The Emma evaluation compared conditions (1) and (3) in terms of perceived usefulness but not directly in terms of task efficiency. It is interesting to ask if this A/B comparison should have been made. For example, the RADAR project measured the performance of a human–agent pair, with and without learning, against that of humans using commercial off-the-shelf tools [30,73].

There were a number of reasons we decided against such a direct comparative study. First, expert experimental design opinion from the HCI community advised us that an A/B study of Emma versus a commercial tool such as Outlook would be unlikely to yield meaningfully reliable data on task effectiveness, due to the difficulty of ensuring comparable tasks between the tools, control of multiple parameters, equivalent user populations, and paradigms of use [22,34]. This is in line with the findings of Höök [39]. Other user-adaptive calendaring assistants would not provide comparable ecological baselines either, since they address distinctly different tasks. Moreover, the learning problems addressed by other adaptive calendaring assistants are sufficiently distinct from PTIME's learning goals that the former fail to provide a useful baseline system (Lesson 1).

The second reason that a direct comparative study was judged inappropriate was the integral role of the preference model in PTIME's creation and exploration of the search space: PTIME was performing a *generative* task [51] Without a preference model, the performance of the constraint reasoner is greatly affected, rendering insignificant any ablative comparison against PTIME without any learning (Lessons 2 and 3). The third reason was that our goal was not to prove the superiority of a research prototype compared to commercial systems, but to validate the promise of the PTIME concept. We wanted to assess whether PTIME is effective at its purpose, whether it is usable for naive users, and what are the subjective opinions of the AI-empowered concept it embeds. Nonetheless, although we did not perform comparative studies, our structured evaluation did include subjective assessment of the perceived usefulness of Emma in relation to existing tools, with the favourable results we reported earlier in Sect. 4.3.

Data and usability. Granted the first control condition of an adaptive system, the perennial issue arises of how to obtain adequate data for the system's learning and for its evaluation. Evaluating an adaptive capability requires data and, for personalized adaptation, as in PTIME—i.e., adaptation to the preferences and practices of an individual—the data must be from, or at least derived from, real users (Lesson 4). (See below for discussion of evaluation realism.) As for PTIME, the data may well have to be collected over time [39], and perhaps from implicit user actions rather than explicit feedback.[1]

Once any system is put in front of users, its usability becomes a crucial issue. If the user cannot or will not interact effectively with the system then the evaluation of the adaptive approach is problematic (Lesson 6). Further, if the task being performed is a *generative* task—requiring interaction with the user to formulate the instance of the task, to which the system generates a response [51]—then the evaluation requires a system with which the user can actually interact and use (Lesson 5); otherwise, no meaningful data can be obtained [76].

Realism of evaluation setting. Consider the behaviour of users interacting with a personalized adaptive system that is performing a generative task, such as PTIME. Users' behaviour is likely to change when the usage situation departs from real life. The issue here is not just

[1] While PTIME can be seen as a type of recommender system, evaluating a task-oriented adaptive system such as PTIME differs significantly from evaluating a classical recommender system, due to the generative, incremental, and dynamic nature of the recommendation task.

that real life has variability that is difficult to capture with a few test cases, but that the very behaviour itself changes—as we saw in the Year 2/3 Test. As we discovered, the complex social and personal factors affecting such interactions cannot always be successfully captured in a controlled environment. Further, our experience with synthetic problems in the Year 4 evaluation of PTIME indicates that it is easy to underestimate the complexity of creating suitable problems for the controlled experiment.

There is a continuum over the level of realism of an evaluation setting: starting at one end, theoretical algorithmic experimentation, to simulation, to controlled 'in-the-lab' studies (with real users), to 'remote' evaluation studies performed by real users but in somewhat less controlled settings (e.g., a Mechanical Turk evaluation), or true 'in-the-wild' studies such as A/B studies or data mining of system usage information. PTIME evaluation involved increasingly realistic evaluation settings as the project progressed, culminating in Year 5's fully 'in-the-wild' study.

Aylett et al. [4] advocate that "there is no substitute" for the process of field studies and trial deployments, and we agree, especially for user-adaptive systems. From the PTIME experience, we conclude that evaluating the usefulness of such systems requires a *deployed* system in real-life use (Lesson 4). Once a system is thus deployed 'in-the-wild' , usability becomes even more crucial (Lesson 6). Further, real users have limited understanding of AI (learning) technology and limited patience in providing explicit feedback and training data [65].

None of this is to say, however, that controlled lab studies are without value. Kjeldskov and Skov [45] point out that laboratory studies can reveal many fundamental limitations of a system, and this was the case with PTIME. Such studies can also 'stress test' a system by requiring users to perform unusually difficult tasks that may occur infrequently in reality (and hence be missed by field studies of limited duration). Theoretical, laboratory, and field studies have complementary roles in gaining a holistic picture of a system.

5.1 Published evaluations of knowledge systems

We remark on three areas of evaluation case studies related to our work. The greatest body of literature on adaptive knowledge system evaluation is perhaps in intelligent tutoring systems: Greer and Mark [35] survey evaluation methods.

First, of the previous explorations around developing user-adaptive calendaring assistants [6,16,49,50,55,56,61,66,70], few have involved actual user studies. Moreover, there are distinct differences between these calendaring assistants that determine what evaluation is appropriate. Brzozowski et al. [16] did conduct a user study evaluating scheduling with the *groupTime* system versus scheduling through email (an A/B evaluation). They found promising results for the usefulness of the tool but did not evaluate the usefulness of the adaptive capability. Refanidis and Alexiadis [66] evaluated the *SelfPlanner* system through an 'in-the-wild' user study of usability, usefulness, and concept validation. The system blended calendaring with to-do task planning but was not adaptive. Schaub et al. [69] evaluated the *PriCal* ambient calendar display through a limited deployment studying social implications, acceptance, and utility. None of these works evaluated the evaluation.

Second, an example of assistive agent technology, the Electric Elves project [19] was taken live at the US government agency DARPA as 'Travel Elves' [2]. Like CALO, the target population was knowledge workers. Travel Elves could, for instance, send faxes to request modification of hotel reservations if a flight was delayed. The deployment was more 'in-the-wild' than PTIME-4 or -5, as the assistive agent system was deployed in an entirely separate organization. As with CALO, significant engineering effort was required and a set

of pragmatic issues had to be addressed, including communication security and privacy. Although the 'elf' agents did not embody nearly as much adaptive capability as CALO, the outcomes from the Travel Elves deployment are relevant to user-adaptive agents. Knoblock [47] reports that after an initially smooth training of users and roll-out of the technology: "agents would fail to execute their assigned tasks, information got delayed or was not sent at all, and in some cases agents failed for no apparent reason". After some initial use, the evaluation experiment was terminated.

Third, a further perspective on the evaluation problem emerges outside the arena of pure academic research. In the commercial arena, there is more often a practical question at the heart of technology evaluation [33,38,68]: a point to which we return to at the end of this article. To assess return on investment, management wants to know whether the value of the personalized adaptive capability is measurable. Practical—and sometimes political—questions concern whether the product will sell, and whether inclusion of (AI) technology will increase customer satisfaction, user productivity, team collaboration, or product differentiation.

Here, Microsoft's ill-fated 'Clippy' has become a classic case study of the transition of an (adaptive) knowledge system. The Microsoft Office Assistant arose from the research-oriented Lumiere project [40]. Commercial time pressure and managerial decisions led to the implementation of a highly restricted subset of the technology. The researchers noted their concern over the lack of usability and user controllability of the resulting system [44]. In this respect, the transitions arising from CALO were much more successful [Fig. 3 (right)].

6 Conclusion and lessons learned

This article contributed a case study in evaluation of a user-adaptive knowledge system. We provided the first systematic account of an evaluation over the extent of the seminal and groundbreaking CALO project. Evaluation of our knowledge system, PTIME, brought forth particular managerial and technical challenges. These came from the nature of the system's personalized assistive task, and from the context of the impact on task performance from learning being the success metric for the larger CALO project. We offer concluding remarks and a set of project-independent take-home lessons in this section.

This article is written from the perspective of hindsight. Technologists such as the majority of the authors are trained primarily to conduct such 'in-the-lab' evaluations. If we had been able to evaluate the concept, the user experience, and the technology from the beginning of the project, and if we had benefited throughout from best-practice guidelines on evaluation from the start, then some of the problems encountered during the project likely could have been anticipated or avoided. As Stumpf et al. [76] put it, "Meaningful interaction involves both parties [system and users] inseparably: approaches must be viable and productive for both the user and the system". Indeed, this is one of the main lessons from our case study of knowledge system evaluation: that technology researchers and project managers can be insufficiently familiar with evaluation methodologies and best practice, and that evaluation is too often included towards the end of a project rather than made an integral part of the development cycle.

Issues that we encountered during the project—and which still arise today— include some long-standing considerations in HCI and experimental design, such as the potential for the think-aloud protocol to interfere with the task being measured, the use of artificial scenarios versus realistic tasks, and the difficulty in defining successful completion of a task [21,52].

Further, our experience with PTIME underlines how it is often difficult to separate out the effects of specific technologies from the overall system performance. Moreover, as remarked earlier, users themselves are often heavily influenced by interface issues, and often cannot distinguish the interface from the performance of the underlying technology.

6.1 Critique and improvement in the PTIME evaluation

Overall, our evaluation efforts for PTIME had mixed success, and we find that the evaluation methodology of CALO in general and PTIME in particular was exposed to flaws. We reiterate that the focus of this article on evaluating the adequacy of the evaluation methodology and drawing lessons from the case study, rather than on the 'success' of the evaluation or the measured knowledge system itself.

First, the evaluations were arguably successful in gaining insight into PTIME (in the form of Emma at the end of the project) regarding usability and perceived usefulness. They were, however, less successful in assessing the learning ability of PTIME, for the reasons discussed in Sect. 4. We can say that the results concerning learning were encouraging in the timescale of adaptation but the evidence tentative. Limited results showing the effectiveness (or otherwise) of a system's learning impacts project planning for the development of a user-adaptive system, since limited data make planning a system development roadmap more difficult.

Second, decisions by the project sponsor significantly impacted the evaluation methodology. The decision to use an independent evaluator had obvious benefits but brought the consequent problems with selecting and biasing Test questions and answers; the decision over the particular evaluation metrics to be used and the somewhat arbitrary targets made the CALO Test results difficult to interpret and skewed development plans towards the Test. If the aim of the CALO-wide evaluation had been to evaluate the usefulness and usability of the technology, the design of the Test would have been different. Third, the subjects in the CALO Test were recruited without specific consideration on the requirements of evaluating PTIME. Further, if CALO were developed in a commercial setting emphasizing usability, subjects would have been new users (as was the case for the longitudinal study of Emma). Related, fourth, the mandated decision to evaluate PTIME only as a part of CALO and at specific points in time did not allow resources to be allocated for (also) conducting independent evaluations throughout. Further, fifth, the initial managerial focus solely on learning to the neglect of usability meant that the project little considered users and use cases . Use cases were successfully developed for the latter technology transitions [12,28,65].

As discussed in Sect. 4.3, despite these problematic aspects, we hold that the experimental methodology for PTIME reached towards the end of the project was appropriate. That section outlined the main points of an ideal evaluation plan, which would have included experiments designed to evaluate system usefulness from the start of the project. Each decision over the evaluation methodology involved a complicated trade-off determining the focus of research and the success criteria for the project, and often subject to CALO-wide factors outside of our remit. These managerial considerations included the scope of the overarching project and its research objectives, the evolution of the project objectives and requirements over time, and the purpose, design, and conduct of CALO-wide evaluations.

As we noted, the years since the conclusion of the project have judged CALO to be a success. Although the CALO Test methodology was imperfect, it did prove to the satisfaction of the project sponsor that the system's task performance increased as a result of machine learning. Moreover, the project arguably attained its broader aim of stimulating research, and—although primarily an academic research effort—from a product perspective, CALO

led to multiple successful spin-out companies. In its domain of personal time management, PTIME had similar success [6,11,67,70]

6.2 Specific lessons highlighted by this case study

Building then on prior analyses of evaluation of intelligent knowledge systems—such as [23,29,37,55,73,76,77], even if these analyses are often of *non*-adaptive or non-personalized systems—six recognized lessons stand out from this case study in the time management domain. These take-home lessons are as relevant today as during the CALO project. We elaborate on each in turn, in the light of the PTIME experience.

The contexts—of the use of technology, and of the competing interests of the stakeholders—must be a primary focus in designing an evaluation strategy. Outside the arena of pure academic research, the overarching question is usually whether a system is more or less beneficial with the technology. There is often a practical question at the heart of the evaluation. Will the product sell? Will customer satisfaction be increased? Will the resulting system reduce errors or increase user productivity? Can we measure the added value of personalized adaptation? Different stakeholders—users, researchers, technologists, companies, funding agencies, the media—often care about different questions and hold different evaluation criteria. Hence, in designing the evaluation of user-adaptive systems, it is paramount to consider the contexts in which the technology is to be used, the perspectives and the relative importance of the stakeholders, and, consequently, the questions the evaluation must answer [34]. Is evaluation to quantify the contributions of academic research? To support a decision to fund product development? To justify deployment? Further, for a user-adaptive system, the way that the answers to these questions are delivered must come with an explanation of what adaptivity means and of the challenges of assessing its impact on users. Whether the primary concern is research or system development/deployment, an adequate evaluation must consider context.

Evaluating one component based on an evaluation of a whole system can be misleading, and vice versa. A layered evaluation of a user-adaptive system seeks to evaluate the system as a whole, and then evaluate the adaptive part [14]. It can mislead to draw conclusions about any one component (e.g., PTIME) based only on evaluation of the whole system (e.g., CALO), but also to evaluate the whole system (CALO) based only on the evaluation of each component alone.

In our case study, the ambition of the annual CALO Test was for a holistic evaluation of the integrated system, in regard to a single criteria of learning performance as a personalized office assistant. For the reasons discussed in Sect. 4, we conclude that the Test was inadequate to evaluate the learning performance of PTIME as one component of the system. For instance, the CALO interface inhibited the use of PTIME and prevented more data from being collected, while the Test and its scoring process more than not failed to accurately measure the performance of PTIME. The evaluation methodology we moved to in the later years of the project was to separate PTIME from the overall system and create a dedicated evaluation plan. This might not be possible for other systems, and regardless one must ask whether it is meaningful to consider a component in isolation.

User-adaptive systems require distinct evaluation strategies, and these strategies must include users. Researchers are trained to evaluate algorithms and technological approaches, primarily to show that they improve performance in some way (e.g., increased predictive accuracy,

precision/recall, efficiency). However, when the objective is to use technology to assist users by adapting to their preferences, theoretical learning curves do not suffice. We evaluated the PTIME preference learner on simulated data and verified its acquisition of scheduling preferences; however, much more effort was needed to evaluate PTIME's learning with user data. In user-adaptive systems, users respond both to the assistance and to the adaptation; it is moot to evaluate learning performance without thought to overall system usefulness [75,78], focusing on realistic tasks in the context of the whole system.

In-the-wild evaluation is necessary when factors affecting user behaviour cannot be replicated in a controlled environment. A key question that must be resolved early in the design process is what points on the continuum—that stretches from algorithmical experiment to evaluation 'in-the-wild' with real users—are most appropriate. With knowledge systems, it can be tempting to think that a highly controlled study will suffice. Controlled evaluations are easier to perform and often yield a greater amount of data in a shorter period of time. However, our experience is that it is all too easy to miss critical, complex social and personal factors surrounding the use of a personalized system. Indeed, these complex factors affecting interactions between a user and other users or agents cannot always be successfully captured in a controlled environment. For PTIME, we found that users behave differently when their decisions do not impact their actual time commitments, requiring that our evaluation could not be conducted wholly in the laboratory. As another example, the power relationships between event initiator and invitees can be influential in the negotiation process, but such social factors can be suspended or changed by a lab setting. In contrast, when the recommendation task is separable from the learning task, or when training data can be effectively gathered offline, meaningful, isolated evaluation of the adaptive capabilities can be conducted.

In-the-wild evaluation implies significant additional development and evaluation costs. To prove return on investment, the costly realism of 'in-the-wild' evaluation is more compelling than 'in-the-lab' evaluation, but conducting such an evaluation is significantly more demanding. Besides care in the selection of the test subject population, attention must be given to system usability, stability, training, and support. The costs of evaluation further increase for a user-adaptive system because of the additional complexity in user and system behaviours.

Even large, well-funded projects such as CALO [58], RADAR [30], and Electric Elves [19] have conducted evaluations that are arguably at best *partially* 'in-the-wild'. As we described, CALO relied on a dedicated critical learning period during which participants conducted loosely scripted office activities with their CALO agents. RADAR devised an artificial conference organization task within which to evaluate the system's ability to help users cope with email overload. Electric Elves was evaluated through actual use over several months—but by its own researchers and with assessment only in terms of indirect metrics, such as the reduction in the number of emails exchanged regarding meeting delays.

In PTIME, we saw how lack of robustness and poor usability can hinder data collection even in a controlled setting. Therefore a strategy is needed to ensure that any system being developed is (1) sufficiently robust to work reliably for the duration of the evaluation; (2) usable and effective enough to be accepted by the subjects; and (3) integrated into the ecology of their working environment.

Ease of adoption of the system by users will determine the success or failure of a deployed evaluation strategy. Users who download a new app will as quickly discard it the first time that they encounter serious bugs or encounter difficulty in accomplishing a task they expect to be 'easy'. Unless the assistance is sufficient (i.e., capable enough), adaptation is irrelevant.

Hence, beyond the maturity and usability of a system, another barrier to adoption may be the paradigm shift implied by new capabilities; barriers may be as much social as technical [4]. A strategy to help avoid this problem is to augment an existing system instead of building an entire replacement. However, augmenting existing systems has challenges, including adequate access to their internals, project constraints, engineering effort, and the suitability of the interaction paradigm. As discussed earlier, PTIME was initially implemented as an add-on to Microsoft Outlook, but project, integration, and evaluation constraints required a PTIME-specific interface to be built.

When building new systems, consideration must be given to users' current work practices and interoperability with current systems and platforms, since change to familiar mission-critical systems can be costly and difficult. We found that even motivated volunteers who disliked their current systems found it hard to remember to use PTIME regularly. Even though a majority of users agreed that scheduling using desiderata constraints is preferable to manually finding times that work for all participants, it was difficult to break the established social practice of sending an email or poll requesting available times as the first step to scheduling. Whether the intent is for shorter-term evaluation or long-term use, ensuring adoption requires the value of change to be demonstrated to stakeholders. Whereas technology—or data-gathering—push is readily felt, user pull can be elusive.

6.3 Final remarks

In conclusion, our experience with evaluation was interwoven with obstacles caused by the user-adaptive nature of the knowledge system being developed, the dynamic nature of the application domain, and the project requirements. Analysing these challenges brought insight not only into the PTIME system as a case study, but also underscored lessons for evaluation of user-adaptive systems more broadly.

Although we chose to evaluate finally the described technology within a stand-alone system, delivery of new technology either embedded within or augmenting existing systems can provide a baseline for evaluation that offers advantages over evaluating a new system. Either way, we conclude that thorough evaluation of an adaptive knowledge system should be multi-faceted and assess—as applicable—the system in terms of usability, usefulness, acceptance, trust, and adaptiveness. Adopting appropriate techniques , the evaluation should be contextualized within the broad problem or research question at hand. We believe that the success of spin-outs from the CALO project has been partly because of these lessons learned for development and evaluation as reported here.

Acknowledgements We thank the anonymous reviewers for suggestions that helped to refine this article. We thank Karen Myers and Daniel Shapiro for their constructive comments, and we thank Mark Plascencia, Aaron Spaulding, and Julie Weber for help with the user studies and evaluations. We thank other contributors to the PTIME project, including Cory Albright, Emma Bowring, Michael D. Moffitt, Kenneth Nitz, Jonathan P. Pearce, Martha E. Pollack, Shahin Saadati, Milind Tambe, Joseph M. Taylor, and Tomás Uribe. We also gratefully acknowledge the many participants in our various studies, and the larger CALO team. For their feedback we thank among others Reina Arakji, Bijan Azad, Jane Davies, Nitin Joglekar, and Alexander Komashie, and the reviewers at the IAAI'09 conference where preliminary presentation of part of this work was made [8]. NYS thanks the Operations group at the Cambridge Judge Business School, where the body of the article was written, the fellowship at St Edmund's College, Cambridge, and the Engineering Design Centre at the University of Cambridge. This material is based in part upon work supported by the US Defense Advanced Research Projects Agency (DARPA) Contract No. FA8750-07-D-0185/0004. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of DARPA, or the Air Force Research Laboratory.

Appendix 1: CALO Test scoring method

The annual CALO Test proceeded as follows, for project Years 2–4 (Sects. 4.1, 4.2).

Methods. The CALO Test process consisted of five parts, as follows for PTIME.

1. The independent evaluator (IE) defined a set of *parameterized questions* (PQs). These templates were made known to the PTIME team, who worked to develop the system's capabilities towards them. There were some 60 PQs relevant to time management. For example:

   Rank the following times in terms of their suitability for [MEETING-ID], given the schedules and locations of anticipated participants.

   Each PQ was supplemented by an agreed interpretation (i.e., what the PQ means) and an ablation process (i.e., how to remove any learning in LCALO, to yield BCALO), both approved by the IE.
2. Data was collected during the week-long critical learning period (CLP).
3. The IE selected a subset of the PQs for evaluation. In Year 2, nine PTIME-relevant PQs were selected. In Year 3, two additional questions were selected.
4. The IE instantiated each PQ with three instantiations relevant to the data set. For example, one instantiation of the above PQ is:

   Rank the following times in terms of their suitability for MTG-CALO-0133, given the schedules and locations of anticipated participants: (1) 7 am, (2) 10 am, (3) 10:30 am, (4) 3 pm, (5) 8 pm.

5. The IE scored LCALO and BCALO on each such *instantiated question* (IQ) instance and produced the overall results. First, the IE determined the 'gold-standard' answer for each IQ. For each PQ, the process for determining the answer key was documented prior to the Test. For example, for the above PQ:

   Since this PQ is not asked from a single user's perspective but from a global perspective (what is best considering all invitees), the Test evaluators will select an arbitrator who will be given access to all calendars and user preferences. The arbitrator may also ask any user for any information that may help the arbitrator identify the best answer. For example, the arbitrator may ask how important the meeting is to a user. The arbitrator will come up with the correct answer.

   While some PTIME IQs had objective answers, others (such as the above) had subjective answers. The IE followed the answer determination process to derive the answer key for each IQ. If necessary, the IE elicited information from the CLP participants, and if further warranted, made subjective final decisions. Second, the IE scored LCALO and BCALO against the answer for each IQ. Scores were between 0 (worst) and 4 (best). Again, for each PQ the process for determining the score was documented prior to the Test. For our example PQ, the process was to compare the ordered list generated by PTIME with the ordered list of the answer key by (quoting verbatim):

   Kendall rank correlation coefficient (also called Kendall Tau) with a shifted and scaled value: Kendall Tau generates numbers between −1.0 and 1.0 that indicate the correlation between two different rankings of the same items. 1.0 indicates the rankings are identical. −1.0 indicates that they are the reverse of each other. Kendall Tau accommodates ties in

the ranking. To get values that range from 0 to 4 (rather than -1.0 to 1.0), we use the following adjustment: Score $=$ (Kendall Tau $+1$) $\times 2$

This scoring process was encoded programmatically so that scores could be computed automatically for LCALO and BCALO.

Tasks. Participants used CALO as part of their regular work activities during the period of the CLP. The participants pursued realistic tasks, in a semi-artificial scenario, i.e., in a dedicated physical location rather than their usual workplaces (Lessons 3 and 4). Participants were given guidance about activities to try to include in their work, for instance, to schedule and attend a certain number of meetings; the independent evaluator approved the guidance. Participants were informed that the CALO system was being evaluated (and not their work) and that they might encounter bugs in the system, due to it being in on-going development.

Appendix 2: Specific critique of the CALO Test

Further to the discussion of Sect. 4.1—the CALO Test aimed for objectivity, as far as could be attained, in providing a quantitative measure of the effects of learning on CALO's performance. However, the nature of the scoring process of the CALO Test introduced unintended artefacts.

First, instantiated questions (IQs) were instantiated (from the parameterized questions (PQs)) with a range of 'difficulty', determined by what the Independent Evaluator (IE) considered easy/difficult for a human office assistant. What is easy or difficult for an intelligent assistant can differ from what is easy or difficult for a human.

Second, as described in "Appendix 1", some IQs had subjective 'gold-standard' answers that required ex-post (i.e., after the activity) elicitation from subjects by the IE, and a partially subjective human decision on the answer key. More generally, a difficulty in evaluation is in defining successful completion of a task. It is worth noting how the PQs were defined by the IE to scope the information required to determine the answer key. For instance, it was not necessary to determine whether users had chosen the best schedules for their requirements out of all possible schedules, but only from the multiple choices of the IQ answers.

Third, for PQs asked as a multi-choice question, a chance effect could unintentionally favour BCALO. For example, consider a multiple choice PQ with two possible answers, A or B, and its three instantiations to IQs. Suppose BCALO has a naive strategy of always returning answer A. There is a $\frac{3}{8}$ probability that for two of the three instantiations, A is the correct answer. In this case, BCALO scores 67% (2.67/4.0), which is higher than the LCALO target for the question!

Fourth, the scoring process for some PQs created artefacts. For example, consider the PQ of "Appendix 1", which was scored using a shifted Kendall rank correlation coefficient. If CALO's answer showed no correlation with the correct answer, it still gets 2 out of 4 points; thus BCALO scored at least 2. Only failing to pick some answer from the given list of choices would score 0 points.

Fifth—a point which we recognized with PTIME, when for instance memory usage of other components slowed CALO's responsiveness—even though the Test was intended to measure CALO's learning ability, it could not do so unless the other parts required by the Test process were in good working order, so that learning data could be collected for LCALO. Architecture, documentation, pretesting, debugging, usability, and user behaviour were therefore all important to scoring well, as much as learning algorithms (Lesson 6).

As a rule, it is difficult in any evaluation to eliminate effects such as selection bias, experimenter bias, learning effects, and the Hawthorne effect— although proper experimental design can minimize them or at least make their effects measurable. The CALO Test was in part deliberately not structured and conducted to eliminate such effects as much as it otherwise might have been, since the CLP was more a data-gathering exercise on the system than a regular user study. For example, whether it was impactful or not upon the data collected, there was selection bias from using subjects from our own institution (which was required for legal reasons). The Test was overseen by the IE, and monitors from the project sponsor were present. Both were satisfied by the validity of the Test results.

References

1. Ackerman S (2011) The iPhone 4S' talking assistant is a military veteran. Wired, 2011. www.wired.com/2011/10/siri-darpa-iphone/. Retrieved 26 Jan 2015
2. Ambite JL, Barish G, Knoblock CA, Muslea M, Oh J, Minton S (2002) Getting from here to there: Interactive planning and agent execution for optimizing travel. In: Proceedings of fourteenth conference on innovative applications of artificial intelligence (IAAI'02), pp 862–869
3. Ambite J-L, Chaudhri VK, Fikes R, Jenkins J, Mishra S, Muslea M, Uribe T, Yang G (2006) Design and implementation of the CALO Query Manager. In: Proceedings of eighteenth conference on innovative applications of artificial intelligence (IAAI'06), pp 1751–1758
4. Aylett R, Brazier F, Jennings N, Luck M, Nwana H, Preist C (1998) Agent systems and applications. Knowl Eng Rev 13(3):303–308
5. Azvine B, Djian D, Tsui KC, Wobcke W (2000) The intelligent assistant: an overview. In: Intelligent systems and soft computing: prospects, tools and applications. Lecture notes in computer science, vol 1804. Springer, New York, NY, pp 215–238
6. Bank J, Cain Z, Shoham Y, Suen C, Ariely D (2012) Turning personal calendars into scheduling assistants. In: Extended abstracts of twenty-fourth conference on human factors in computing systems (CHI'12)
7. Berry PM, Gervasio M, Peintner B, Yorke-Smith N (2007) Balancing the needs of personalization and reasoning in a user-centric scheduling assistant. Technical note 561, AI Center, SRI International
8. Berry PM, Donneau-Golencer T, Duong K, Gervasio MT, Peintner B, Yorke-Smith N (2009a) Evaluating user-adaptive systems: lessons from experiences with a personalized meeting scheduling assistant. In: Proceedings of twenty-first conf. on innovative applications of artificial intelligence (IAAI'09), pp 40–46
9. Berry PM, Donneau-Golencer T, Duong K, Gervasio MT, Peintner B, Yorke-Smith N (2009b) Mixed-initiative negotiation: facilitating useful interaction between agent/owner pairs. In: Proceedings of AAMAS'09 workshop on mixed-initiative multiagent systems, pp 8–18
10. Berry PM, Gervasio M, Peintner B, Yorke-Smith N (2011) PTIME: personalized assistance for calendaring. ACM Trans Intell Syst Technol 2(4):40:1–40:22
11. Bosker B (2013a) Tempo smart calendar app boasts Siri pedigree and a calendar that thinks for itself. The Huffington Post. www.huffingtonpost.com/2013/02/13/tempo-smart-calendar-app_n_2677927.html. Retrieved 30 June 2016
12. Bosker B (2013b) SIRI RISING: the inside story of Siri's origins—and why she could overshadow the iPhone. The Huffington Post. www.huffingtonpost.com/2013/01/22/siri-do-engine-apple-iphone_n_2499165.html. Retrieved 10 June 2013
13. Bosse T, Memon ZA, Oorburg R, Treur J, Umair M, de Vos M (2011) A software environment for an adaptive human-aware software agent supporting attention-demanding tasks. Int J Artif Intell Tools 20(5):819–846
14. Brusilovsky P, Karagiannidis C, Sampson D (2004) Layered evaluation of adaptive learning systems. Int J Contin Eng Educ Lifelong Learn 14(4–5):402–421
15. Brusilowsky P (2001) Adaptive hypermedia. User Modell User Adapt Interact 11(1–2):87–110
16. Brzozowski M, Carattini K, Klemmer SR, Mihelich P, Hu J, Ng AY (2006) groupTime: preference-based group scheduling. In: Proceedings of eighteenth conference on human factors in computing systems (CHI'06), pp 1047–1056
17. Campbell M (2009) Talking paperclip inspires less irksome virtual assistant. New Scientist, 29 July 2009
18. Carroll JM, Rosson MB (1987) Interfacing thought: cognitive aspects of human-computer interaction. MIT Press, Cambridge

19. Chalupsky H, Gil Y, Knoblock CA, Lerman K, Oh J, Pynadath DV, Russ TA, Tambe M (2002) Electric elves: agent technology for supporting human organizations. AI Mag 23(2):11–24
20. Cheyer A, Park J, Giuli R (2005) IRIS: integrate, relate, infer, share. In: Proceedings of 4th international semantic web conference on workshop on the semantic desktop, p 15
21. Christie CA, Fleischer DN (2010) Insight into evaluation practice: a content analysis of designs and methods used in evaluation studies published in North American evaluation-focused journals. Am J Eval 31(3):326–346
22. Cohen P (1995) Empirical methods for artificial intelligence. MIT Press, Cambridge
23. Cohen P, Howe AE (1989) Toward AI research methodology: three case studies in evaluation. IEEE Trans Syst Man Cybern 19(3):634–646
24. Cohen PR, Howe AE (1988) How evaluation guides AI research: the message still counts more than the medium. AI Mag 9(4):35–43
25. Cohen PR, Cheyer AJ, Wang M, Baeg SC (1994) An open agent architecture. In: Huhns MN, Singh MP (eds) Readings in agents. Morgan Kaufmann, San Francisco, pp 197–204
26. Cramer H, Evers V, Ramlal S, Someren M, Rutledge L, Stash N, Aroyo L, Wielinga B (2008) The effects of transparency on trust in and acceptance of a content-based art recommender. User Model User Adap Int 18(5):455–496
27. Davis FD, Bagozzi RP, Warshaw PR (1989) User acceptance of computer technology: a comparison of two theoretical models. Manag Sci 35:982–1003
28. Deans B, Keifer K, Nitz K et al (2009) SKIPAL phase 2 final technical report. Technical report 1981, SPAWAR Systems Center Pacific, San Diego
29. Evers V, Cramer H, Someren M, Wielinga B (2010) Interacting with adaptive systemsInteractive collaborative information systems, volume 281 of studies in computational intelligence. Springer, Heidelberg
30. Freed M, Carbonell J, Gordon G, Hayes J, Myers B, Siewiorek D, Smith S, Steinfeld A, Tomasic A (2008) RADAR: a personal assistant that learns to reduce email overload. In: Proceedings of twenty-third AAAI conference on artificial intelligence (AAAI'08), pp 1287–1293
31. Gena C (2005) Methods and techniques for the evaluation of user-adaptive systems. Knowl Eng Rev 20(1):1–37
32. Grabisch M (1996) The application of fuzzy integrals in multicriteria decision making. Eur J Oper Res 89(3):445–456
33. Graebner ME, Eisenhardt KM, Roundy PT (2010) Success and failure in technology acquisitions: lessons for buyers and sellers. Acad Manag Perspect 24(3):73–92
34. Greenberg S, Buxton B (2008) Usability evaluation considered harmful (some of the time). In: Proceedings of twentieth conference on human factors in computing systems (CHI'08), pp 111–120
35. Greer J, Mark M (2016) Evaluation methods for intelligent tutoring systems revisited. Int J Artif Intell Educ 26(1):387–392
36. Grudin J, Palen L (1995) Why groupware succeeds: discretion or mandate? In: Proceedings of 4th European conference on computer-supported cooperative work (ECSCW'95), pp 263–278
37. Hall J, Zeleznikow J (2001) Acknowledging insufficiency in the evaluation of legal knowledge-based systems: Strategies towards a broad based evaluation model. In: Proceedings of 8th international conference on artificial intelligence and law (ICAIL'01), pp 147–156
38. Hitt LM, Wu DJ, Zhou X (2002) ERP investment: business impact and productivity measures. J Manag Inf Syst 19:71–98
39. Höök K (2000) Steps to take before intelligent user interfaces become real. Interact Comput 12(4):409–426
40. Horvitz E, Breese J, Heckerman D, Hovel D, Rommelse K (1998) The Lumière project: Bayesian user modeling for inferring the goals and needs of software users. In: Proceedings of 14th conference on uncertainty in artificial intelligence (UAI'98), pp 256–266
41. Jameson AD (2009) Understanding and dealing with usability side effects of intelligent processing. AI Mag 30(4):23–40
42. Joachims T (2002) Optimizing search engines using clickthrough data. In: Proceedings of 22nd ACM conference on knowledge discovery and data mining (KDD'02), pp 133–142
43. Kafali Ö, Yolum P (2016) PISAGOR: a proactive software agent for monitoring interactions. Knowl Inf Syst 47(1):215–239
44. Kahney L (2010) MS Office helper not dead yet. Wired, 19 April 2001. www.wired.com/science/discoveries/news/2001/04/43065?currentPage=all. Retrieved 8 Oct 2010
45. Kjeldskov J, Skov MB (2007) Studying usability in sitro: simulating real world phenomena in controlled environments. Int J Hum Comput Interact 22(1–2):7–36
46. Klimt B, Yang Y (2004) The Enron corpus: a new dataset for email classification research. In: Proceedings of 15th European conference on machine learning (ECML'04), number 3201 in lecture notes in computer science. Springer, pp 217–226

47. Knoblock CA (2006) Beyond the elves: making intelligent agents intelligent. In: Proceedings of AAAI 2006 spring symposium on what went wrong and why: lessons from AI research and applications, p 40
48. Kokalitcheva K (2015) Salesforce acquires "smart" calendar app Tempo, which is shutting down. Fortune. www.fortune.com/2015/05/29/salesforces-acquires-tempo/. Retrieved 30 June 2016
49. Kozierok R, Maes P (1993) A learning interface agent for scheduling meetings. In: Proceedings of international workshop on intelligent user interfaces (IUI'93), pp 81–88
50. Krzywicki A, Wobcke W (2008) Closed pattern mining for the discovery of user preferences in a calendar assistant. In: Nguyen NT, Katarzyniak R (eds) New challenges in applied intelligence technologies. Springer, New York, pp 67–76
51. Langley P (1999) User modeling in adaptive interfaces. In: Proceedings of 7th international conference on user modeling (UM'99), pp 357–370
52. Lazar J, Feng JH, Hockheiser H (2010) Research methods in human–computer interaction. Wiley, Chichester
53. Maes P (1994) Agents that reduce work and information overload. J ACM 37(7):30–40
54. McCorduck P, Feigenbaum EA (1983) The fifth generation: artificial intelligence and Japan's computer challenge to the world. Addison Wesley, Boston
55. Mitchell T, Caruana R, Freitag D, McDermott J, Zabowski D (1994) Experience with a learning personal assistant. Commun ACM 37(7):80–91
56. Modi PJ, Veloso MM, Smith SF, Oh J (2004) CMRadar: a personal assistant agent for calendar management. In: Proceedings of agent-oriented information systems workshop (AOIS'04), pp 169–181
57. Moffitt MD, Peintner B, Yorke-Smith N (2006) Multi-criteria optimization of temporal preferences. In: Proceedings of CP'06 workshop on preferences and soft constraints, pp 79–93
58. Myers KL, Berry PM, Blythe J, Conley K, Gervasio M, McGuinness D, Morley D, Pfeffer A, Pollack M, Tambe M (2007) An intelligent personal assistant for task and time management. AI Mag 28(2):47–61
59. Nielsen J, Levy J (1994) Measuring usability: preference vs. performance. Commun ACM 37(4):66–75
60. Norman DA (1994) How might people interact with agents. Commun ACM 37(7):68–71
61. Oh J, Smith SF (2004) Learning user preferences in distributed calendar scheduling. In: Proceedings of 5th international conference on practice and theory of automated timetabling (PATAT'04), pp 3–16
62. Oppermann R (1994) Adaptively supported adaptivity. Int J Hum Comput Stud 40(3):455–472
63. Palen L (1999) Social, individual and technological issues for groupware calendar systems. In: Proceedings of eleventh conference on human factors in computing systems (CHI'99), pp 17–24
64. Paramythis A, Weibelzahl S, Masthoff J (2010) Layered evaluation of interactive adaptive systems: framework and formative methods. User Model User Adap Interact 20(5):383–453
65. Peintner B, Dinger J, Rodriguez A, Myers K (2009) Task assistant: personalized task management for military environments. In: Proceedings of twenty-first conference on innovative applications of artificial-intelligence (IAAI'09), pp 128–134
66. Refanidis I, Alexiadis A (2011) Deployment and evaluation of Selfplanner, an automated individual task management system. Comput Intell 27(1):41–59
67. Refanidis I, Yorke-Smith N (2010) A constraint-based approach to scheduling an individual's activities. ACM Trans Intell Syst Technol 1(2):121–1232
68. Rychtyckyj N, Turski A (2008) Reasons for success (and failure) in the development and deployment of AI systems. In: Proceedings of AAAI'08 workshop on what went wrong and why: lessons from AI research and applications, pp 25–31
69. Schaub F, Könings B, Lang P, Wiedersheim B, Winkler C, Weber M (2014) PriCal: context-adaptive privacy in ambient calendar displays. In: Proc. of sixteeth international conference on pervasive and ubiquitous computing (UbiComp'14), pp 499–510
70. Shakshuki EM, Hossain SM (2014) A personal meeting scheduling agent. Pers Ubiquit Comput 18(4):909–922
71. Shen J, Li L, Dietterich TG, Herlocker JL (2006) A hybrid learning system for recognizing user tasks from desktop activities and email messages. In: Proceedings of eighteenth international conference on intelligent user interfaces (IUI'06), pp 86–92
72. SRI International (2013) CALO: cognitive assistant that learns and organizes. https://pal.sri.com. Retrieved 10 June 2013
73. Steinfeld A, Bennett R, Cunningham K et al (2006) The RADAR test methodology: evaluating a multi-task machine learning system with humans in the loop. Report CMU-CS-06-125, Carnegie Mellon University
74. Steinfeld A, Bennett R, Cunningham K, et al. (2007a) Evaluation of an integrated multi-task machine learning system with humans in the loop. In: Proceedings of 7th NIST workshop on performance metrics for intelligent systems (PerMIS'07), pp 182–188

75. Steinfeld A, Quinones P-A, Zimmerman J, Bennett SR, Siewiorek D (2007b) Survey measures for evaluation of cognitive assistants. In: Proceedins of 7th NIST workshop on performance metrics for intelligent systems (PerMIS'07), pp 189–193
76. Stumpf S, Rajaram V, Li L, Wong W-K, Burnett M, Dietterich T, Sullivan E, Herlocker J (2009) Interacting meaningfully with machine learning systems: three experiments. Int J Hum Comput Stud 67(8):639–662
77. Tambe M, Bowring E, Pearce JP, Varakantham P, Scerri P, Pynadath DV (2006) Electric Elves: what went wrong and why. In: Proceedings of AAAI 2006 spring symposium on what went wrong and why: lessons from AI research and applications, pp 34–39
78. Van Velsen L, Van Der Geest T, Klaassen R, Steehouder M (2008) User-centered evaluation of adaptive and adaptable systems: a literature review. Knowl Eng Rev 23(3):261–281
79. Viappiani P, Faltings B, Pu P (2006) Preference-based search using example-critiquing with suggestions. J Artif Intell Res 27:465–503
80. Wahlster W (ed) (2006) SmartKom: foundations of multimodal dialogue systems. Cognitive technologies. Springer, New York
81. Weber J, Yorke-Smith N (2008) Time management with adaptive reminders: two studies and their design implications. In: Working Notes of CHI'08 workshop: usable artificial intelligence, pp 5–8
82. Wobcke W, Nguyen A, Ho VH, Krzywicki A (2007) The smart personal assistant: an overview. In: Proceedings of the AAAI spring symposium on interaction challenges for intelligent assistants, pp 135–136
83. Yorke-Smith N, Saadati S, Myers KL, Morley DN (2012) The design of a proactive personal agent for task management. Int J Artif Intell Tools 21(1):90–119

Pauline M. Berry is Senior Director of Analytics and Data Engineering at xAD, Inc. Previously she was Senior Director of Analysis at Yahoo!, Inc., and worked on the CALO project while at SRI International. She holds a doctorate from the University of Strathclyde.

Thierry Donneau-Golencer is Director of Product at Salesforce, Inc. He co-founded Tempo AI, a spin-off from the CALO project at SRI International acquired by Salesforce in 2015. He holds a masters degree in Artificial Intelligence from Oregon State University.

Khang Duong is a Software Engineer at SRI International.

Melinda Gervasio is a Principal Scientist at the Artificial Intelligence Center, SRI International. Her research interests include intelligent assistants, mixed-initiative systems for planning and scheduling, adaptive personalization, and end-user programming. Her current work uses AI in human–machine team settings with the goal of building collaborative autonomous systems. She holds a doctorate from the University of Illinois at Urbana–Champaign.

Bart Peintner is co-founder and CTO at Loop AI Labs, Inc, working to radically change how machines can autonomously learn and understand the human world. Previously he was a Senior Computer Scientist at the Artificial Intelligence Center, SRI International. He holds a doctorate from the University of Michigan.

Neil Yorke-Smith is an Associate Professor at the Suliman S. Olayan School of Business, American University of Beirut. His research interests include planning and scheduling, agent-based methodologies, data analytics, simulation, and their real-world applications. He holds a doctorate from Imperial College London.