Complex Systems Informatics and Modeling Quarterly
CSIMQ, Article 74, Issue 13, December 2017/January 2018, Pages 1–21
Published online by RTU Press, https://csimq-journals.rtu.lv
https://doi.org/10.7250/csimq.2017-13.01
ISSN: 2255-9922 online

Capacity Management as a Service for Enterprise Standard Software

Hendrik Müller*, Sascha Bosse, and Klaus Turowski

Faculty of Computer Science, Otto-von-Guericke-University, Magdeburg, Germany

hendrik.mueller@ovgu.de, sascha.bosse@ovgu.de, klaus.turowski@ovgu.de

Abstract. Capacity management approaches optimize component utilization from a strong technical perspective. In fact, the quality of involved services is considered implicitly by linking it to resource capacity values. This practice hinders to evaluate design alternatives with respect to given service levels that are expressed in user-centric metrics such as the mean response time for a business transaction. We argue that utilized historical workload traces often contain a variety of performance-related information that allows for the integration of performance prediction techniques through machine learning. Since enterprise applications excessively make use of standard software that is shipped by large software vendors to a wide range of customers, standardized prediction models can be trained and provisioned as part of a capacity management service which we propose in this article. Therefore, we integrate knowledge discovery activities into well-known capacity planning steps, which we adapt to the special characteristics of enterprise applications. Using a real-world example, we demonstrate how prediction models that were trained on a large scale of monitoring data enable cost-efficient measurement-based prediction techniques to be used in early design and redesign phases of planned or running applications. Finally, based on the trained model, we demonstrate how to simulate and analyze future workload scenarios. Using a Pareto approach, we were able to identify cost-effective design alternatives for an enterprise application whose capacity is being managed.
Keywords: capacity planning, performance prediction, response time, server consolidation, utilization, optimization, enterprise applications.

1 Introduction

The performance of enterprise applications (EA) is critical to the successful execution of corporate business functions and business tasks [1]. Therefore, it must not be degraded significantly [2] in order to support business processes effectively. Negative consequences of performance failures may include damaged customer relations, lost income, increased

* Corresponding author

Reference: H. Müller, S. Bosse, and K. Turowski, "Capacity Management as a Service for Enterprise Standard Software," *Complex Systems Informatics and Modeling Quarterly*, CSIMQ, no. 13, pp. 1–21, 2017. Available: https://doi.org/10.7250/csimq.2017-13.01

Additional information. Author's ORCID iD: H. Müller – orcid.org/0000-0002-5083-536X. Article PII S225599221700074X. Article received: 2017 September 29. Accepted: 2017 December 20. Available online: 2017 December 29.

maintenance costs, delayed project schedules, and project failures [3]. In addition, EAs tend to grow depending on a changing customer base, new product releases, new divisions and acquisitions [1]. Consequently, they need to be monitored and managed continuously and proactively in order to ensure a stable quality of service that is aligned to business requirements [4]. On the other hand, information technology (IT) represents a major cost factor for enterprises. According to the worldwide IT spending forecast by Gartner, overall IT costs will grow by 2.6 percent to a total of 3.55 trillion dollars in 2018. Particularly, costs for enterprise software will grow by 7.1 percent [5]. Hence, IT Service Management (ITSM) frameworks such as ITIL and ISO 20000 embed the task of balancing performance and operational costs into the capacity management process [6]. Since energy costs are a major factor in the total cost of ownership of modern data centers [7], [8], [9], savings can be achieved by avoiding idle resources, which may consume up to 70 percent of their peak power [10]. Therefore, the provisioning of minimum hardware capacity that still ensures cost-effective operations, aligned to given service levels and business constraints, is at the heart of capacity management decisions. For this purpose, differently sized design alternatives need to be evaluated for the EA under study. Capacity can be defined as the maximum throughput an enterprise application, hereafter also referred to as service, can achieve with given response time thresholds for different transaction types [11], [12]. While capacity planning is a crucial task that necessitates to foresee the expected workload of a planned EA, capacity management refers to the optimization of existing EAs [13].

In the latter case, workload data may be available as a result of application performance monitoring (APM) activities and can be used to determine required capacity levels for different time intervals. Capacity management should be a continuous task [4] in order to address existing optimization potential which results from low server utilization levels. According to several studies, average server utilization levels usually vary between 10 and 50 percent [9], [10], [14]. In June 2016, the United States Data Center Energy Usage Report forecasted the average utilization of active volume servers to be approximately 15 % for internal data centers and 25 % for data centers of service providers in 2020 [15]. According to this report, volume servers represent by far the largest share of total servers in data centers and show an adverse power proportionality in the effect of consuming about 50 percent of their maximum power usage at low utilization levels of around 10 percent. Therefore, raising utilization levels barely effects power consumption but enables to shut down idle servers after their running services have been relocated in accordance to a globally optimal distribution. Since only active servers where considered in the report, an additional portion of inactive servers further increases the optimization potential. On this matter, recent studies state that up to 30 percent of all servers operated in the US were not used in 6 months or more and, thus, termed as comatose [16], [17]. Consequently, several approaches exist to address existing optimization potential by consolidating services on a reduced number of servers in accordance to their workload profiles [9]. Typically, such approaches formulate one- or multidimensional optimization problems having one or multiple resource dimensions and, optionally, one time dimension. While the majority of consolidation approaches focuses on demands and capacities in terms of the CPU, few approaches take also main memory consumption into account [18]. Since optimization problems are known to be NP-hard, solution candidates are calculated using metaheuristics or genetic algorithms. We believe that such approaches are an excellent example of creating real business value from interdisciplinary research that addresses requirements from business informatics by applying solution algorithms to a mathematical problem representation. As a matter of fact, the majority of algorithms we studied come, under their individual assumptions, to feasible solutions where expected resource demands of services do not exceed the capacity limits of the assigned servers. However, solution feasibility is usually evaluated from a strong technical perspective, e.g., the fitness function of a genetic algorithm rates a solution candidate's feasibility with respect to capacity savings. As stated by Stillwell et al., "in practice, however, resource management objectives are expressed based on higher level metrics that are related to user concerns, such as service response time or throughput" [19]. Likewise, service level

agreements (SLA) are typically expressed from a user perspective to ensure measurability. Server consolidation approaches consider those higher level metrics only indirectly by mapping them to resource capacity values in a reasonable way [19], e.g., a certain amount of CPU capacity allows for the execution of a certain number of business transactions per hour [20]. Yet this high level abstraction does not allow to qualify response times or other related user-level metrics and strongly limits solution evaluation with respect to given SLAs. Consequently, some approaches account for this aspect by providing means to add spare capacity of up to 50 percent per service in order to ensure acceptable response times [21], [22]. This strategy, in turn, dramatically lowers the addressable optimization potential to an extent that questions the entire consolidation effect. The concept of spare capacity is certainly valid to catch unpredictable load peaks but should not be overused to compensate the missing link between quality of service (QoS) and resource capacity. Therefore, we claim that current approaches lack to directly evaluate identified solutions with respect to QoS aspects. On this matter, an estimation of expected service response times is desirable in order to both increase the addressable optimization potential and to allow for mapping the costs of solution candidates to SLAs. Hence, to support decisions as part of a capacity management process, the performance of an enterprise application needs to be predicted and evaluated considering different design alternatives.

As stated by the authors of [11], ideally, a system comparable to the final production environment would be desirable during capacity testing. However, this precondition is rarely fulfilled and smaller scale systems are hardly comparable [11]. As a result, model-based techniques evolved to assess future system performance and traditional prediction literature heavily relies on queuing networks and discrete event simulation [23]. Unfortunately, such concepts and methods related to performance modeling and analysis are largely unknown to most IT practitioners [24] and, thus, are rarely applied in industry [25]. Main reasons include a generally high time effort in construction and analysis of analytical models [26] as well as essential expert knowledge about the model itself, the system to be implemented and its dependencies which may not be available [27]. In addition, the credibility of obtained results remains questionable until their validation in later lifecycle phases [6]. Therefore, measurement-based approaches are more frequently used in practice than model-based approaches due to their effectiveness in operation [27]. As a costly consequence, performance requirements are considered lately and performance testing is done when an implemented or relocated system is about to go live [28], [29]. This practice was originally termed by [30] as “fix it later approach”, but can be observed in today’s projects either [3]. The correction of performance failures at that last stage is inefficient, expensive, delaying and professionally irresponsible [29], [31]. "We cannot do anything about performance until we have something running to measure" is a statement from practice that was identified by [3] in 2011 as a frequent argument for managing performance reactively.

However, we argue that, due to the intensive use of standard software in the field of enterprise applications [32], [33], there is actually something running to measure, even though it is likely not operated within the enterprise that applies capacity management. If widespread and well-established standard software is utilized, there is a high likelihood that components of the system whose capacity is being managed, are already in production in various environments. Moreover, those components may already produce performance-related log data as part of APM activities [13], [34], [35], [36]. Thus, measurement data are present allowing for measurement-based techniques to be used for performance assessment in the design or redesign phase of planned and existing enterprise applications. Such data contain information in terms of performance affecting patterns that need to be extracted and processed in order to transfer the implicit knowledge to other environments. For this purpose, machine learning techniques can be utilized in order to design a pure black box approach that does not require domain-specific and potentially expensive expert knowledge about the system structure and behavior. In [6], it was shown that predictions of mean response times per business transaction on a cross-organizational data basis can be as accurate as model-based approaches. However, the authors did not demonstrate how to

utilize the prediction results in the context of planning and managing the capacity of enterprise applications with respect to costs. Main activities in this field such as the analysis of future scenarios and the prediction-based decision support were not yet considered. Apart from that, many efforts have been conducted to describe those activities in a systematic way. One particular capacity planning process that is consistently being used and adapted [11], was developed by [37], although it was originally designed for web services.

In this article, based on the groundwork of [6], we designed a multi-organizational capacity management process that is particularly beneficial for users of enterprise standard software. While based on the main activities of the process by [37], it integrates specifications from the domain of enterprise applications with regard to their common characteristics. Furthermore, we added activities from the domain of data science to ensure a structured usage of machine learning techniques while building standardized performance models. Those activities are carried out by an internal or external capacity management provider who collects and processes measurement data from various running EAs in order to build prediction models for each standard software component. This way, predictions can be consumed as a service using shared models, leveraging economies of scale and reducing capital costs of the capacity management consumer. After models were trained and successfully evaluated, they allow predicting service performance for different future scenarios. Within this process activity, we integrated the concept of Pareto-optimal solutions to identify cost-effective design alternatives and provide convenient decision support. The presented research extents our prior work which was published in [38].

The remainder of the article is structured as follows. After an initial discussion on the state of the art with respect to capacity management and prediction techniques, we summarize results of a field study that examines the average standardization degree of usage for enterprise standard software in Section 2. Having formed this basis of our work, the designed capacity management process for enterprise standard software is being presented in Section 3. According to the design science paradigm [39], [40], we evaluate the artefact in Section 4 through demonstration, where we particularly focus on innovative aspects such as the integration of machine learning elements. Using a real-world example, we utilize measurement data from 1,821 different instances of enterprise standard software to train a performance model. Subsequently, we demonstrate how to apply the model to predict service performance for forecasted workloads of a managed enterprise application using alternative designs. To complete the evaluation, we analyze potential future scenarios and derive decision support for our practical example. The findings are summarized in Section 5 where we also discuss the approach and clearly point out existing limitations. To address the latter, future research topics are identified.

2 State of the Art

Capacity management is a challenging and often complex task [9], [41], hence, many efforts have been made to structure required activities, to automate sub-processes and to support decisions. Especially, if the level of complexity requires to utilize artificial intelligence in order to effectively solve certain subtasks, different research fields are affected. In the following, we provide a short overview of the state of the art in the fields of capacity management as well as performance prediction techniques. Furthermore, we present practical insights about the average standardization degree of usage for EAs.

2.1 Service and Component Capacity Management

EAs need to permanently adjust to the individual, ever-changing business environment which they support, resulting in a continual requirement for tuning and sizing [1]. Therefore, application demands have to be aligned with the available hardware resources on a periodical basis. This challenge is at the heart of any capacity management effort. To ensure an adequate use of available capacity in practice, regular server consolidation projects are carried out by

internal or external service providers. The objective of corresponding activities is to determine a cost-effective input of hardware resources that enables a satisfying output of operations in the sense that given service level agreements are principally not violated. Besides the generally preferred use of energy-efficient hardware, this can be achieved by placing services on available servers such that idling capacity is avoided at most times. Hence, tasks and objectives of the ITIL sub-processes *service capacity management* and *component capacity management* are equally involved.

For EAs, the workload often follows seasonal patterns on a daily and weekly basis [22], [23], [42] while changes appear only slowly in the long term [41]. Therefore, workload profiles, containing resource demands for different points in time, can be derived from historical data for each service. These profiles are used to make well-founded placement decisions where complementary workloads can be operated together in order to increase the overall average server utilization. Naturally, many approaches exist to address this optimization potential. The described challenge is closely related to the well-known bin packing problem, which is NP-hard. Hence, usually a single- or multi-dimensional workload consolidation problem is formulated and metaheuristics or genetic algorithms are utilized in order to identify feasible solutions to the problem. Herein, mainly CPU and, optionally, memory demands represent the considered resource dimensions [18]. However, as user-centric metrics such as service response times are not considered in the problem formulation, the quality of service for the newly calculated allocations remains questionable before its actual deployment. As argued in Section 1, predictions of to be expected performance would be desirable in order to foresee the effects of deploying a designed solution. In [18], some excellent work related to the descried optimization problem were studied and classified regarding different aspects of their application area. Following our argumentation, we further investigated the listed publications with respect to the evaluation of QoS aspects and summarize the results in Table 1.

Table 1. Classification of related work

Publication	Workload	Service Distinction	Quality of Service
Rolia2003 [21]	dynamic	no	spare capacity
Rolia2005 [22]	dynamic	yes	spare capacity
Bichler2006 [23]	dynamic	no	not explicitly considered
Cherkasova2006 [41]	dynamic	yes	spare capacity
Stillwell2010 [19]	static	no	not explicitly considered
Xu2010 [66]	static	no	not explicitly considered
Speitkamp2010 [14]	dynamic	no	quantiles of historical service demands
Feller2011 [43]	dynamic	no	not explicitly considered
Gao2013 [67]	static	no	not explicitly considered

At first, Table 1 distinguishes between static and dynamic workload. If the optimization problem implicates a time dimension for service capacity demands, we consider the approach to be dynamic, following the classification by [43]. In contrast, static approaches usually take the peak demand of a service and optimize allocations with respect to this single value, thus, practically wasting optimization potential. Furthermore, different service types may have different QoS requirements. While performance for productively operated ERP systems must not be degraded significantly [10], a temporary loss of performance can be tolerated in some cases, e.g., for development instances in favor of overall costs. Therefore, the assignment of service types allows for a more granular service treatment with regard to individual performance requirements. Finally, we studied the problem formulation for each approach and examined whether QoS aspects were explicitly considered. In the following we summarize the results for those publications that considered QoS aspects at least indirectly.

Rolia et al. (2003) seek to a soft assurance for quality of service by introducing an additional portion of unused capacity of 50 percent or 20 percent per CPU or server [21]. In 2005, Rolia et

al. distinguish two service classes for production and non-production workloads. Furthermore, they define differing resource access probabilities for interactive and batch workloads to ensure that degraded resource access will not result in higher application response times. During evaluation, they use a spare CPU capacity of 50 percent of the targeted utilization for all interactive workloads in order to keep their response times low [22].

The approach of Cherkasova et al. (2006) allows application owners to define utilization ranges of acceptable and degraded performance for a defined percentage of measurements in the historical workload trace. This way, the amount of spare capacity can be adjusted for specific services and time intervals [41].

In 2010, Stillwell et al. state that higher level metrics can be mapped to resource fractions allocated to a service by means of a linking metric. They create a metric, yield, which quantifies for each service how much of the resource demand is actually satisfied. In their allocation problem, they maximize the minimum yield over all services to consider performance and fairness. However, actual response time estimations were not quantified [19]. In the same year, Speitkamp et al. state that workloads which exceed the available capacity may lead to increased response times, but it remains unclear to which extent response times are affected [14].

To conclude, all studied approaches consider the QoS at best only indirectly by adding spare capacity for certain workload types or time intervals so that resource overloads are avoided. Nevertheless, none of the approaches quantifies user-level metrics such as the expected service response times for the identified solution candidates. Instead, performance-related metrics are abstracted by means of other metrics that are based on the resource utilization. While the provisioning of spare capacity certainly helps to catch unexpected load peaks, this strategy, depending on the added percentage, may dramatically reduce the optimization potential.

Furthermore, it remains unclear how much spare capacity needs to be added in order to ensure a desired service level. This decision seems to be based on experience, the risk attitude of the application owner, and the consequential costs of service level violations. Hence, to quantify an adequate capacity amount, expert knowledge is required, which, on the one hand, may be costly or not available, and, on the other hand, implies the risk of a *rather too much than too little*-attitude, especially if a majority of the considered services has strict performance requirements. Ultimately, additional capacity of the same resource type may not decrease response time to a desired service level. In fact, the performance of services also benefits from certain resource types or architectures that can be integrated by means of performance models [6].

To summarize, the prediction of performance appears to be decoupled from service placement problems in the literature while their strong interdependence to support business functions cost-effectively is often disregarded. From a user perspective, pure landscape optimizations with the objective to maximize the average server utilization on the basis of historical workloads provide a proper starting point for placement decisions but still require to integrate means of verifying the expected QoS. We argue that approaches similar to the studied publications are suitable to be extended by subsequent performance prediction techniques, since related metrics are frequently collected as part of instrumented APM facilities [1], [13] and, therefore, may already be included in the workload traces of the underlying capacity management data basis. Hence, prediction models can be trained using workload traces and utilized to predict service response times on targeted servers before the actual deployment of any service relocation. We further note that approaches which use dynamic workload profiles are best suited to be integrated with prediction models because response times depend on metrics that usually vary for each measured point in time such as the number of active users, the number of database requests or the amount of transferred data between the layers of the EA [6]. If similar metrics are available and linked to services and timestamps, capacity management benefits from an integrated knowledge base that holds monitoring data for server consolidation efforts as well as the training and use of prediction models.

2.2 Prediction Models in Decision Making

An integral part of the designed capacity management process is to evaluate alternatives in decision making. According to the intelligence-design-choice model by [44], three fundamental phases are required to support decisions:

- Intelligence phase: understand the problem.
- Design phase: generate and evaluate alternatives.
- Choice phase: compare alternatives and decide.

In simple environments, decisions can be taken automatically if the effects of alternatives are previously known. In all other cases, the effects of decision alternatives should be predicted with a suitable accuracy. In that regard, the use of prediction models allows to map decision variables to objective values. In the following, we distinguish between qualitative and quantitative prediction approaches. Qualitative approaches focus on the comparison of alternatives using language-like results, e.g., "better than" [45]. Often, subjective methods such as expert interviews are utilized to evaluate alternatives. Therefore, such approaches are easy to understand and to apply, but limited in credibility and transferability. On the other hand, quantitative approaches focus on the objective assessment of a single alternative. Here, two general modeling approaches exist [25], namely white-box (also referred to as model-based) techniques as well as black-box (also referred to as measurement-based) techniques [46]. The former describe the internal structure and behavior of alternatives on the basis of common models. According to a "divide and conquer strategy", the problem is decomposed into subproblems until solutions to subproblems are easy to obtain. Such approaches rely on high modeling effort, but may produce comprehensible results since the model behavior can be analyzed. White-box models can be evaluated analytically and by simulation, prototyping, or testing [25]. Examples include process algebra, stochastic processes, Petri nets, queuing networks (QN), business process modeling notation (BPMN) or unified modeling language (UML) extensions [28].

In contrast, black-box approaches describe parametric models that can map the relation between input and output values on a high level. The alternatives are modeled as input-output tuples so that machine learning techniques can be used to train prediction models. While no domain experts need to be consulted for modeling, massive amounts of data may be required. Therefore, in the field of capacity management, implemented systems may be needed to produce measurement data for model training [28]. While the accuracy can be evaluated easily and, if appropriate, results are credible, trained models may not be comprehensible since general and transferable knowledge cannot be easily obtained from internal model characteristics [47]. Examples include regression models, artificial neural networks, support vector machines, and random forests.

While many research activities were conducted using white-box approaches to model planned enterprise applications [11], only few attempts exist to use machine learning within capacity planning or management. In [48], a regression-based analytical model was trained to forecast the workload evolution. However, to predict actual service performance a QN was used. Recent machine learning approaches in this field mainly aim at integrating and analyzing datacenter logs across different layers of a data center to allow for predictive maintenance and root cause analyses in case of component failures [49]. Such concepts are referred to as IT Operations Analytics (ITOA), but do typically not consider the cross-organizational integration of measurement data collected from one particular layer in order to identify performance patterns. A first proof of concept for software-specific performance predictions using a black-box model that was trained across organizational borders was published by [6]. We build upon these results and construct an extended capacity management process for enterprise applications that takes account for the wide dissemination of standard software in this domain. Since the process builds on the assumption of functional components being comparable across different instances of the

same software, we performed a study on the average usage degree of standard business transactions for a major EA product, which we present in the following chapter.

2.3 Standardization Degree of Enterprise Applications

In the field of EAs, standard software is intensively utilized [32], [33]. However, every standard software needs to be adapted to fit the individual organizational and operational structure of its target environment. As part of this customization process, nonexistent but required functionality is provided by means of creating custom business transactions. Their execution logic is unique and typically unknown to external organizations. Therefore, corresponding performance metrics are hardly comparable across different instances of an EA [6]. Since the designed approach relies on standardized performance models, it may not be cost-effective for enterprise applications that utilize custom transactions to a great extent. Instead, economies of scale can only be leveraged for business transactions that are offered by the vendor of the EA to a large customer domain.

In order to practically assess the effects of this limitation on a standardized capacity management approach, we performed a field study across 1,299 enterprise data centers with the objective to gather insights about the average standardization degree of productively operated EAs. For a period of averagely three weeks per data center, we observed a total of 176,782,062 dialog transaction calls across 11,626 instances of the enterprise resource planning (ERP) standard software SAP ERP, regardless of the branch or country it was operated in. In average, 18.03 % of these executions were related to custom transactions, leading to an average standardization degree of approximately 82 % across the analyzed data centers. Figure 1 shows the data distribution by means of a histogram.

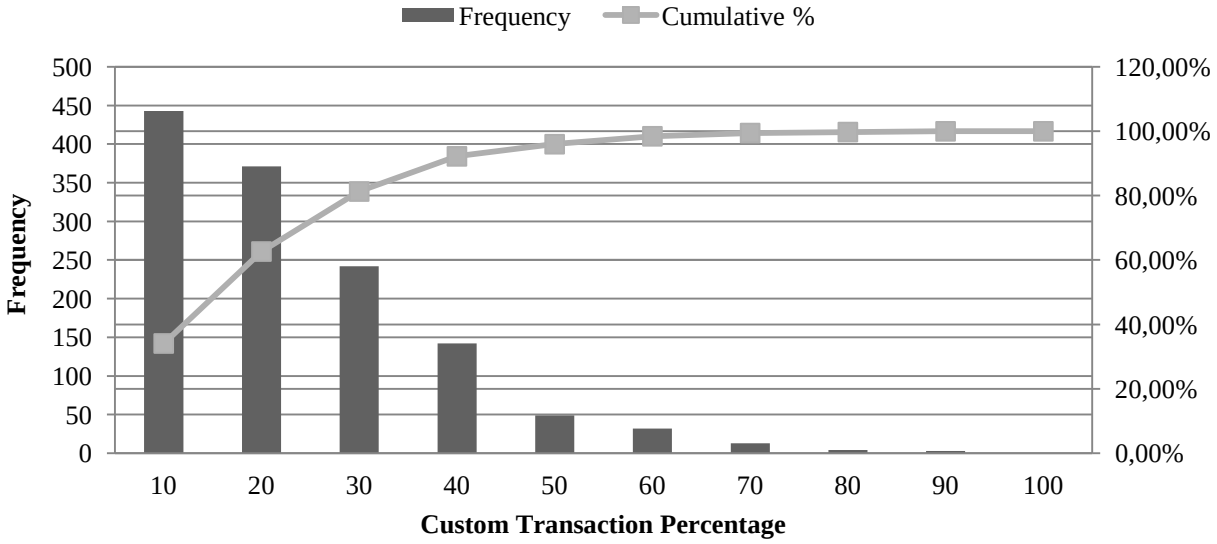

Figure 1. Usage degree of custom transactions for an enterprise standard software

As can be seen in Figure 1, within the largest class of data centers (443 out of 1.299), less than 10 % of executions were related to custom business transactions. This accounts to 34 % of the analyzed data centers. In general, 81.3 % of the data centers show a custom transaction percentage of less than 30 % which is relatively close to a 80/20 Pareto distribution. Furthermore, 96 % of the data centers spent less than 50 % of the dialog executions on custom transactions. This practical study supports the theoretical assumption that ERP implementation projects aim at using as much of the existing functionality as possible without having to customize [50]. Hence, the average usage profile of an EA strongly fosters the feasibility of a domain-specific but standardized capacity management approach that can be offered as a service in order to leverage economies of scale.

3 Capacity Management for Enterprise Standard Software

IT professionals employ capacity management activities to ensure an acceptable QoS for systems under study [24], [29], [37], [51]. As discussed in the previous section, performance measurements can be useful for capacity management, although a massive amount of training data is required due to the black-box nature of this approach. Therefore, a running system or parts of it must be available [6]. In the domain of enterprise applications, organizations heavily utilize standard software along with its economies of scale, e.g., for ERP systems provided by software vendors to a large number of organizations. Especially large-scale information systems are rarely developed completely from scratch. Instead, most applications are constructed by adapting existing software to new organizational contexts [32]. In fact, over 60 % of the U.S. Fortune 1000 utilize third-party standard software provided by few major ERP vendors [33]. As a consequence, enterprise software to be introduced is most likely already running in different production environments and associated standard transactions are consistently being executed. These applications typically come with integrated software monitors that measure performance at runtime as part of APM activities. Since measurements can be extracted from a production system [13], performance-related data of systems comparable to the one that is being planned may be present in different organizations. As stated by [13], data collected by APM tools, e.g., about EA topologies, transaction traces, and performance measures, are a valuable input for performance model extraction approaches and may replace assumptions with knowledge. Since APM data includes transaction usage, resource utilization, system topology and response times, it can be used for performance prediction of standard transactions by utilizing machine learning techniques [6]. Such models, trained on a large volume of performance logs, recognize existing patterns, e.g., across hardware capacities, software releases and response times and, therefore, may be utilized to simulate effects of planned capacity changes such as the relocation of services as a result of server consolidation projects. Hence, we state that existing capacity management approaches do not leverage effects of the wide dissemination of enterprise standard software and waste collaboration potential. To address this gap, we modeled a capacity management process that allows leveraging those aspects. As stated in Section 1, we build upon the capacity planning process designed by [37], although it was initially defined only for web services. Since design requirements for enterprise applications differ from generic internet applications in many aspects [1], we further specified the process for EAs. As an example, business dynamics affect both the number of active users in an enterprise application and the number and extent of database requests [1]. Hence, the capacity needs to be aligned with short- mid- and long-term requirements determined by the business strategy. A major adaption we made to the process by [37] is to utilize machine learning techniques for performance model training. Therefore, well-established concepts from the broad field of data science must be considered to ensure that models are systematically built and evaluated. Thus, we integrated tasks from the knowledge discovery in databases (KDD) process [52] into the capacity management process. In [6], a domain-specific performance knowledge base was proposed to support performance predictions for standard business transactions. The capacity management provider utilizes the concept of such a knowledge base in order to accomplish data storage, analytics and provisioning of results.

3.1 Process Overview

The capacity management process aims at identifying cost-effective design alternatives, e.g., in terms of hardware configurations, that allow for maximizing performance while minimizing costs. Since some of the main steps of the capacity management process were already introduced by [37], we focus on changes we made to adapt the process with regard to the specific characteristics of enterprise standard software. Figure 2 illustrates the process, which was modeled using BPMN 2.0. We chose this language since capacity management needs to be easily understood and integrated into application lifecycles [1] to be utilized by practitioners.

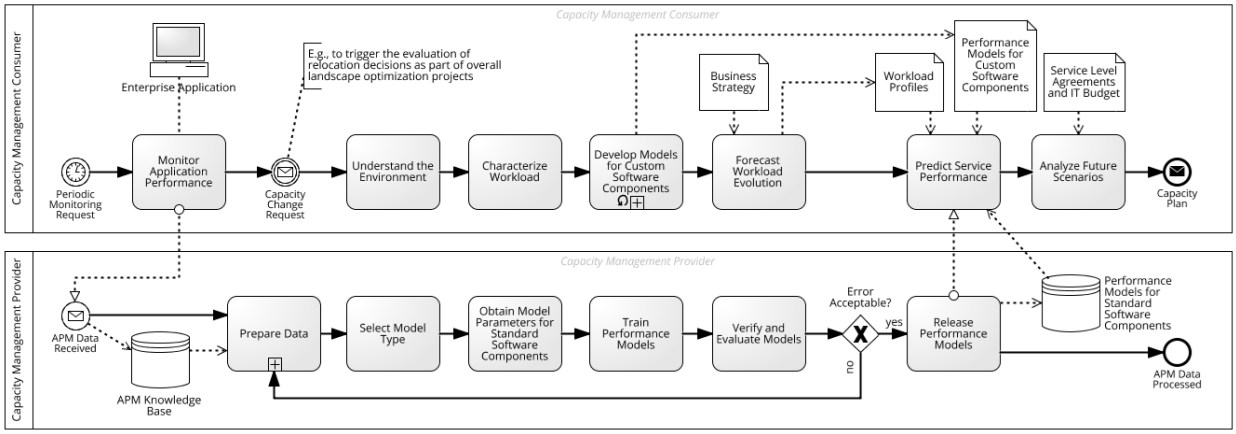

Figure 2. Capacity management process model for enterprise standard software

The Capacity for EAs is typically planned and managed on a transactional basis [11]. Therefore, different prediction models can be built for individual business transaction types in order to simulate their performance. We distinguish between standard and custom transaction types. If customers require functionality beyond available components, the source code of the product must be adapted [50], resulting in custom transaction types that are known solely to the organization and are not used by any other party. Therefore, cross-organizational performance models cannot be built for these components and their capacity needs to be planned traditionally using individual analytical models or gathered data. However, a field study we performed prior to this work (cf. Section 2.3) reveals an average standardization degree of approximately 72 % across 11,626 instances of the enterprise standard software SAP ERP. Hence, the majority of transaction executions produces measurement data that is comparable across organizational borders [6] and, therefore, can be integrated to train standardized prediction models. To support both customized and standard transaction types to be considered, two organizational entities are involved, represented by the horizontal swim lanes (cf. Figure 2):

- **Capacity management consumer**: IT organization that hosts the enterprise application whose capacity is intended to be planned or managed.
- **Capacity management provider**: an entity that integrates measurement data from different application environments and provides prediction models for standard software components. This can be either the software vendor itself or a (internal or external) third party who provides capacity management as a service.

The capacity management consumer carries out APM activities in order to collect the necessary data basis. This happens either continuously, e.g., by means of software intrusion, or on a periodical basis after a representable time frame was chosen. Since the APM data is at the heart of any data-driven capacity management approach, such traces are transferred to the capacity management provider, who seeks to keep performance models up to date.

As indicated by the intermediate message start event, the actual capacity management process is triggered whenever a capacity change is required. Since the performance of an EA needs to be continuously evaluated [11], current capacity plans must undergo an evaluation process as performance changes are detected, e.g., through a continuous application performance monitoring or by end users via an incident management. In such cases, capacity change requests may follow. Besides continuous performance evaluations, the process is being triggered as part of major technical, strategic or organizational changes. Technical changes may be caused by migration, relocation, upgrade or any other change projects that are expected to affect the EA's performance. This includes preceding server consolidation activities that produced new allocations of running services to available servers which need to be verified with respect to the expected quality of service (cf. Section 2.1). Organizational changes, on the other hand, include

openings of new divisions, mergers and acquisitions [1] while strategic changes may result in an expansion of the customer base or the release of new products. Such changes can heavily affect the expected workload. Finally, periodic audits or assessments may reveal either performance problems or optimization potential, of which both should entail capacity change requests. In the following section, we depict the main activities of the process.

3.2 Activities of the Capacity Management Provider

To enable capacity management for standard components, a capacity management provider utilizes the concept of a domain-specific performance knowledge base [6] to store, prepare and analyze a variety of measurement data. The objective of these activities is to provide the capacity management consumer with standardized performance models to be used for performance predictions of alternative scenarios, offered as a service. Therefore, capacity management consumers need to provide monitoring data that originates from APM activities. In the domain of enterprise applications, this is a common practice to allow consultancies for bottleneck identification, performance anomaly detection, performance tuning and related services, e.g., as used by [53]. Within the sub-process “Prepare Data” (cf. Figure 2), the capacity management provider selects, cleans, filters and aggregates the measurement data to fit into the data schema of the APM knowledge base. The data, in any manner, must contain information about the system topology, the resource utilization, and workload characteristics for each given time interval [6]. As part of this task, undesired outliers need to be excluded by means of quantiles and domain experts [54]. As different prediction models can be suitable depending on the transaction type and the data distribution, a suitable model type is to be selected by means of related studies and as part of the result verification loop. Appropriate model types for predicting business transaction performance may be regression trees, random forests, support vector machines, gradient boosting machines or models based on evolutionary learning [6]. The subsequent activity “Obtain Model Parameters for Standard Software Components” identifies and groups performance-related metrics by classes of similar components [37]. In the domain of EAs, such classes can be represented by specific standard business transaction types or standard task types like batch jobs or dialog jobs. To enable model evaluation, data need to be divided into a training and a test subset while the training subset should be at least twice as big as the test subset [55], [56]. After measurement data were preprocessed, they can be used to train the selected type of prediction model. Depending on the amount of training data, the model type and the degree of parallelism, this activity can take up to several hours. However, once a model is trained and provisioned, predictions can be made within seconds. Priory, verifying and evaluating the models is a crucial activity. If a model of insufficient prediction accuracy is being used to simulate alternative scenarios, business decisions are compromised, eventually leading to either serious performance degradations and business disruptions, or to costly investments for underutilized hardware resources. Therefore, the gathered test subset is used during the activity “Verify and Evaluate Models” to evaluate the model accuracy when being used with unknown data. Here, error metrics such as the mean absolute error (MAE) and the mean absolute percent error (MAPE) need to be investigated [57], [58], which should be based on absolute differences to prevent the compensation of positive and negative differences. Both measures are recommended, because the MAE quantifies the error and the MAPE generates a relation to observed values. In general, the boundaries of acceptable values need to be defined by the modeler [37]. As stated by [59], errors up to 25 % are, according to general conditions, within acceptable ranges. Likewise, [51] stated that accuracies of performance models from 10 to 30 % are acceptable in capacity planning. If the error was classified to be not acceptable, parts of the described KDD process need to be repeated. Means of tuning include changing the model parameters, changing the selected dataset, e.g., by applying additional filters, or selecting a different model type. These steps need to be iterated until the error is acceptable, also referred to as calibration procedure [37]. After that, trained prediction models can be provisioned to be

utilized by the capacity management consumer. Finally, provisioned models need to be updated on a regular basis by the provider to reflect additional and changed patterns that are caused by new software releases, new hardware components or any technological paradigm changes.

3.3 Activities of the Capacity Management Consumer

The capacity management consumer generally follows a sub-process that integrates well-known activities such as monitoring the application performance, understanding the environment and characterizing the workload. Detailed descriptions and guidelines related to these tasks can be found in [37]. In the following, we focus on the adoptions we made when it comes to the development and usage of performance models. For custom software components, prediction models need to be developed individually as outlined, e.g., in [11]. For standardized software components, such as basic business transaction types, the capacity management provider has provisioned prediction models to be used by any consumer as described in the previous section. In order to predict their service performance, these models and input data that reflect anticipated workload characteristics are needed. Sources of information can be observations of the system under study, if the capacity of an existing system is being managed, or benchmarking results if the capacity for a new system is being planned [13], [37]. To predict future service performance, input data must be aligned with the profiles of the forecasted workload, which represent the output of the activity "Forecast Workload Evolution". Workload profiles define load-related parameters for a time interval such as the number of active users, the number of business transactions, the number of screen changes (also referred to as dialog steps), the number of database service units [60], and the utilization of the central processing unit (CPU) in terms of normalized metrics. Since EA workloads strongly depend on an ever-changing business environment [1], they must be forecasted on the basis of the business strategy (cf. Figure 1). The time horizon is a major aspect in the forecasting process, where seasonal, trend and random workloads for short-term, intermediate-term and long-term periods can be distinguished [37]. Aspects from the related concept of time series components by [61] may be used additionally in order to take adequate account to business plans and product lifecycles. Accordingly, mid-term developments with variable cycle length are specified as cyclic components whereas short-term seasonal components show regular changes resulting in a constant cycle length. Components may be integrated additively or multiplicatively [61]. Hence, the activity "Forecast Workload Evolution" produces workload profiles for each time horizon that are used in the subsequent activity for predicting service performance (cf. Figure 2). Here, the provisioned performance model for the standard software component of interest is utilized and the identified workload profiles serve as an input to specify model features that need to be adapted for the prediction. The use case and the granularity of available APM data determine the value that is to be predicted. Examples may be

- the mean response time per time interval for each dialog step (DS) of a certain type of standard business transaction,
- the mean response time per time interval for a task type, e.g., for any standard batch job or dialog job of a defined complexity, or
- the throughput of an application instance per time interval.

The response time was defined by [29] as the time a system spends on reacting to a human request. Predictions need to be repeated using the workload profiles and alternative designs in order to create a broad spectrum of future scenarios as a decision basis for the subsequent activity. If the prediction of service performance was requested as part of a server consolidation project, historical workload traces are to be spanned over targeted server configurations. Hence, the prediction models are used with values of the monitored metrics that are linked to new resource capacities, enabling to forecast performance values of a service for each considered point in time. If preceding optimization algorithms have been configured to identify a number of

alternative solution candidates, those serve as model input in order to produce alternative future scenarios.

In any case, the predicted scenarios need to be analyzed by IT decision makers with respect to given service level agreements (SLA) and assigned IT budget. Within the activity "Analyze Future Scenarios", the scenarios are used to limit the solution space of suitable design alternatives to those that meet the given SLAs at lowest costs. According to [37], the analyzed scenarios should consider the expected workload, the costs and the QoS. Consequently, all design alternatives with minimized costs and maximized performance need to be identified, leading to a multi-objective optimization problem. Such problems can be addressed by a Pareto approach as done, e.g., in [62] for optimizing performance and costs and in [63] for optimizing service availability and costs. On this basis, design decisions can be made, resulting in a capacity plan.

4 Evaluation

We demonstrate the feasibility of the designed process using a real-world example of an enterprise application that utilizes widespread standard software shipped by the software vendor SAP. In the context of ongoing server consolidation efforts, we assume a productively operated service to be chosen for a relocation. As multiple design alternatives have been identified, their expected quality is to be verified in order to choose the optimal solution with respect to costs and performance. Enterprise applications such as SAP systems typically support the execution of predefined standard business transactions as well as customized business transactions that were developed in-house. To support capacity management for both custom and standard components, the process branches into two paths as described in Section 3. For the purpose of demonstrating the feasibility of the process, we focus on the artifacts of each path that we designed within the scope of this article and assume the respective entity's role. We use and mention obtained results from other activities if required for the sake of transparency. For custom components, analytical models can be used, which were extensively investigated in scientific literature and, therefore, are not demonstrated during evaluation. Instead, we firstly focus on the activities performed by the capacity management provider utilizing techniques from the field of machine learning as described in Section 3.2. Furthermore, we demonstrate the prediction of service performance for a standard business transaction and the subsequent analysis of alternative future scenarios (cf. Figure 2). Finally, using a Pareto approach, decisions are being supported.

4.1 Data Preparation and Model Selection

The used enterprise application under study is constantly being monitored by the capacity management consumer utilizing software monitors, which in our case are an integral component of the application. In the course of an ongoing server consolidation project, optimization algorithms as introduced in [18] came to a number of alternative designs for the running EAs. All design alternatives would cause reductions of the average server utilization. However, some services require a strict fulfillment of defined performance goals for selected business transactions. Since multiple solution candidates exist for the target design of each service, performance predictions support the final decision with respect to minimized response times and costs. Furthermore, due to planned business changes, the EA's capacity needs to be evaluated for future scenarios of potentially higher workload.

Therefore, most recent monitoring data of the last three weeks were sent to the capacity management provider and loaded into the APM knowledge base, which contains performance observations from various instantiations of this application type. As the capacity management provider, we then grouped these observations by business transaction type and hour. To enable model training for our given example, data tuples referring to more than 60,000 observations of hourly mean response times for a transaction type were utilized. The data layer, the analytics

layer and the provisioning layer of the APM knowledge base were set up as outlined in [6] using an in-memory database system. Therefore, each observation contains information about the workload extent during the observed hour, the system topology including the release status as well as the involved infrastructure components and their utilization levels. As part of the task "Prepare Data", we excluded undesired outliers, e.g., caused by hanging processes, and filtered the data to include only productively used systems and to ensure minimum hardware utilization levels (the final data set comprises measurement data that originates from 1,821 instances of productively used EAs). Subsequently, we selected a model type. In [6], service performance of business transactions were predicted using six different prediction model types, including regression trees, random forests and support vector machines. When those were compared with respect to their mean errors across all cases, best prediction accuracy was achieved using a random forest. Based on these findings, we performed training runs using random forests that combine 1000 individual decision trees.

4.2 Model Training and Evaluation

As part of the activity "Obtain Model Parameters for Standard Software Components" we conducted expert interviews and, hereby, identified 30 parameters of our observations that may influence the quality of service performance, e.g., the number of concurrently active users and the data volume that is transferred between the application server and the database server of the EA. For dividing the data into a training and a test subset, we used a ratio of 80/20. As stated by [37], often differently parameterized training iterations are necessary to achieve a satisfying accuracy. During the runs, we used a feature importance function to assess the mean decrease of accuracy if a feature was permuted before prediction. This way, features having low influence can be identified and excluded from additional iterations. In our case, most important features include the number of database service units and the number of dialog steps performed during the studied hour. In our example, we used training data that is associated with executions of the standard business transaction for changing sales orders. On the basis of the total test set consisting of 12,774 predictions, error metrics have been computed for each prediction run. While the mean response time across all observations accounted to 594.14 milliseconds (ms), a mean absolute error of 124.6 ms was achieved, resulting in a mean absolute percentage error of 22.4 %. The MAE is rather difficult to interpret for a single prediction model, but it could be utilized to compare different prediction models. The MAPE is easier to interpret since it allows for a percentage assessment and, consequently, is more frequently used in business environments [64]. In this context, the MAPE is significantly lower than the proposed accuracy of 25 % [59] and respectively 30 % [51], what is required for prediction models in capacity management. Thus, the trained model can be used in the context of performance prediction for different capacity or load scenarios.

4.3 Service Performance Prediction

The capacity management consumer wishes to predict the mean response time per dialog step for a business transaction type within an hour of runtime that reflects the workload profile identified in the previous activity (cf. Figure 2). In our early work, we introduced optimization algorithms which are capable of solving multidimensional workload consolidation problems [18]. According to Table 1, the optimization approach was implemented to support dynamic workloads and service distinction. In this article, however, we introduce an additional verification of the QoS for the identified design alternatives, since the sole consideration of CPU and memory demands can hardly be mapped to given SLAs. While the foregoing server consolidation is out of scope in this work, we assume that our optimization algorithms came to alternative placement decisions for a productively running service with exceptional significance for business continuity. Therefore, solution candidates need to be verified with regard to their

performance for future scenarios of different load levels. The demonstration of the corresponding workload forecast and derivation of workload profiles was outlined, e.g., in [37]. In our given case, the capacity management consumer expects a seasonal workload as observed in the historical data but with increasing trend. In particular, due to the integration of an additional business unit, the number of active users is expected to double while their task types remain nearly the same. The characterization of workload has revealed that the most used business transaction was to change sales orders, represented by the standard SAP transaction code VA02. Therefore, we predict the mean response time for this transaction type using the provisioned model and load factors between 1 and 2 with regard to the current workload extent. Since the workload characterization revealed several observations, especially beyond office hours, having less user activity, we added load factors between 0.25 and 1. The prediction results are indicated by the line that corresponds to design alternative 1 in Figure 3.

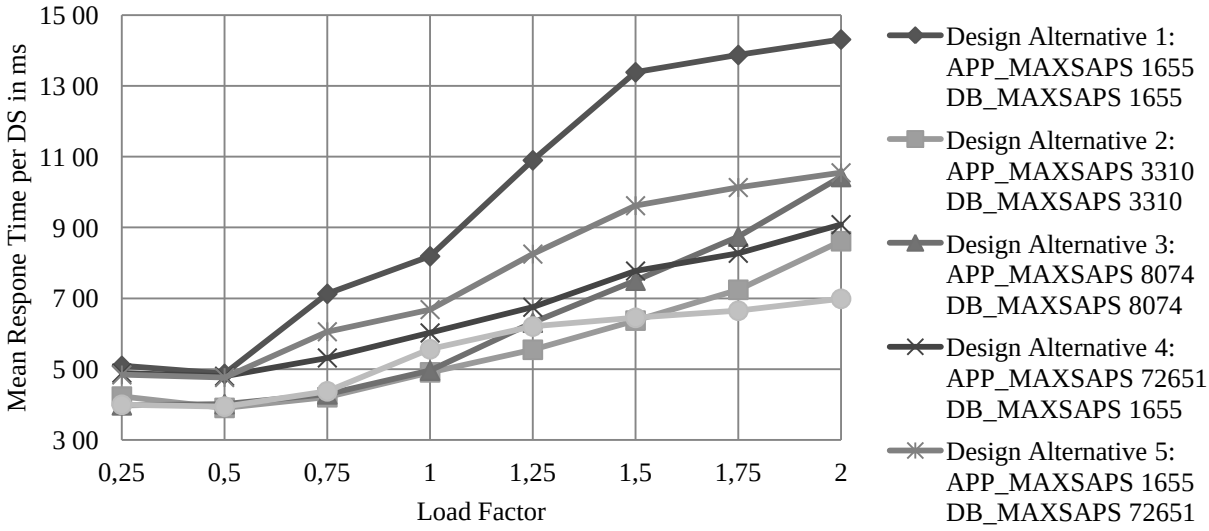

Figure 3. Performance prediction results for alternative scenarios

In our example, the given operational level agreements (OLA) between the business unit and the internal IT service provider do not allow for mean response times per dialog step longer than one second. As an example, the design alternative 1 would cause OLA violations for load factors of 1.25 or higher (cf. Figure 3). In our case, the design alternatives represent real server fractions with different hardware characteristics where their capacity is most relevant for performance. Hardware capacity can be measured in SAPS (SAP Application Performance Standard), where 100 SAPS equal the capability to process 6,000 dialog steps per hour [20]. Database and application layers of EAs often span across multiple physical servers. The system under study consists of two servers for the database and the application instance. Therefore, in Figure 3, we distinguish between these two capacity values for each design alternative. However, other design characteristics such as the CPU type and the number of cores may additionally contribute to performance changes, sometimes resulting in anomalies where higher capacity does not guarantee lower response times (cf. Design Alternative 2 and 3). The predictions were repeated for each design. Due to their perceptible implications on the performance, the presented designs can be evaluated while the accuracy (cf. Section 4.2) must always be considered, especially if prediction results do not vary significantly. For each design, we assign costs amounting to the required total capacity in SAPS as also done by [9]. Therefore, multiple designs would enable to meet the given OLAs at different costs. In order to support investment decisions, it is the subject of the next process activity, “Analyze Future Scenarios”, to identify those design alternatives that deliver highest performance at lowest costs.

4.4 Analysis of Future Scenarios

Considering a high number of possible design alternatives, e.g., due to possible combinations, a decision-maker should not be overwhelmed by alternatives that are not optimal. On the basis of validated performance models, different scenarios can be quantitatively analyzed and compared. Since the minimization of response time as well as costs are conflicting objectives, a Pareto approach is used to address the optimization problem.

In this context, we are interested in the set of non-dominated solutions for all workload levels to be analyzed, i.e., all decision alternatives for which no better response time can be achieved without introducing additional costs.

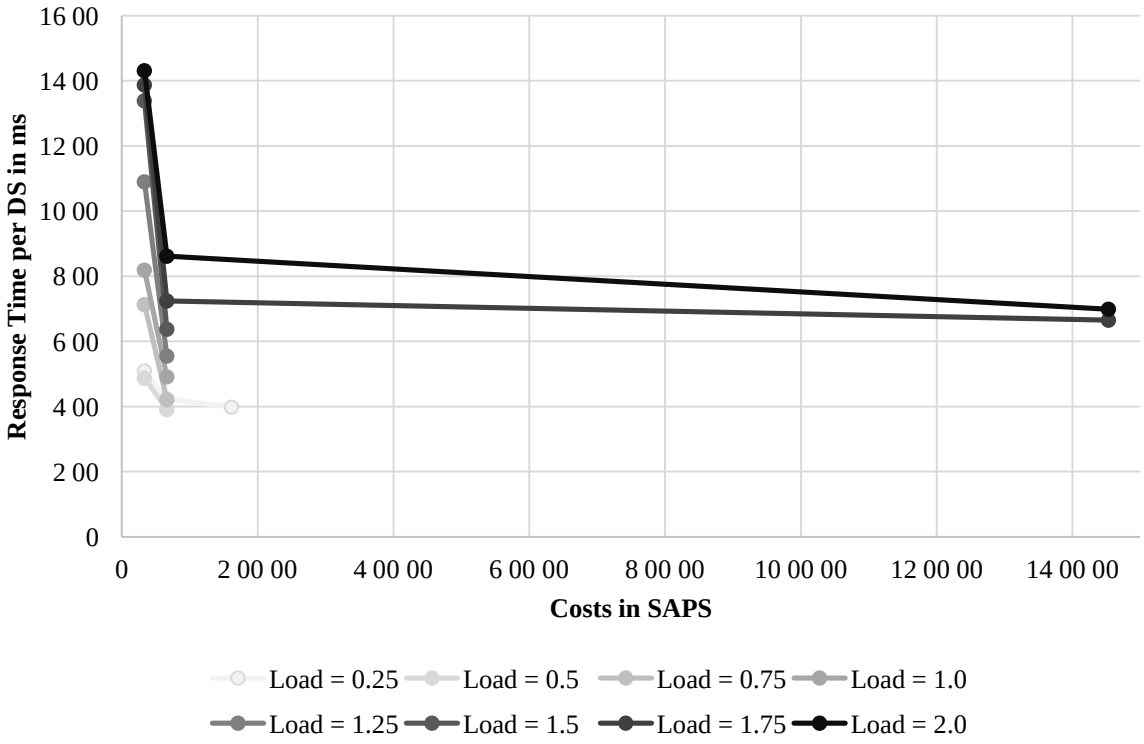

Figure 4. Set of Pareto-optimal designs

In Figure 4, the set of illustrated Pareto fronts indicates the relation between costs and response time for each workload level. Each point column refers to a Pareto-optimal design alternative, i.e., design alternative 1 with capacity costs of 3,310 SAPS, design alternative 2 with 6,620 SAPS and design alternative 6 with 145302 SAPS (cf. also Figure 3). In each column, the single points refer to the different mean response times predicted at different workload levels. On that basis, given OLAs or SLAs can be addressed cost-effectively. If, as indicated in the previous section, a mean response time of 1 second must not be exceeded, the design alternative 1 will likely violate this constraint if the mean workload increases by 25 %. In this case, the design alternative 2 would be necessary which will likely meet the constraint even for a doubled workload. If, however, the mean response time constraint is decreased to 800 ms and workload is doubled in comparison to the mean workload, design alternative 2 will not suffice. Thus, design 6 comprising a high-end application and database server with costs amounting to 145,302 SAPS would be required. To conclude, the final design choice depends on given SLAs and the expected load scenario. This example demonstrates how the management can utilize the information generated in the capacity management process. If, however, the number of possible solution candidates makes an evaluation of all candidates infeasible, (meta-)heuristics, such as the NSGA-II presented by [65], can be applied to explore the search space more efficiently for optimal candidates.

5 Conclusion

ERP products are seen as an opportunity both to share data and to standardize processes across the organization. On the other hand, capacity management for such widely spread systems is still performed highly individually with very few degrees of standardization and collaboration. Capacity management that is provided as a service can leverage economies of scale since performance models for standard software components are reused within the community of a particular EA. In this article, we designed and evaluated a process that benefits from machine learning to support capacity management challenges for consumers who utilize enterprise standard software. Since performance models are trained across measurement data from various running enterprise applications, capacity management benefits from a pure and reliable black-box approach, applicable in the design or redesign phase of an EA and reducing both prediction and correction costs. Using more than 60,000 observations from productively running EAs, a prediction model of satisfying accuracy could be trained. As a potential capacity management consumer, we demonstrated the process activities to utilize trained models for performance predictions and to identify cost-effective design alternatives that consider workload forecasts resulting from the business strategy. In contrast to studied traditional capacity management approaches which optimize resource usage from a strong technical perspective, the designed process supports to quantify and verify the QoS for identified alternative designs. Hence, user-level metrics that can be directly linked to SLAs such as the mean response time for standard business transactions are used to identify a set of pareto-optimal solutions with respect to costs and expected QoS. By transferring the challenges of performance predictions into a service concept, the capacity management consumer can focus on business challenges and decision making without having to deal with the various aspects of data science. Moreover, the designed process enables business models for capacity management on a pay-per-use basis, which may be of particular interest for hardware vendors who offer design alternatives. Finally, the research artefact may affect software selection processes for enterprise applications by contributing an additional decision-making factor to the main question of standard software versus individual software. The concept may influence decisions in favor of standard software to benefit from services that are built upon APM knowledge bases such as the capacity management as a service.

We summarized our results and exposed the value of the designed research artifact. However, existing limitations of the approach need to be considered and addressed by future research activities. Firstly, the accuracy of the prediction models highly depends on the amount of training data and, therefore, on the number of observations that are stored in the APM knowledge base. Consequently, black-box approaches can be applied on standard transaction types that are used by a wide range of ERP customers. Therefore, model evaluation, using the described error metrics, is of major importance to assess the limits of the model's application area with respect to the variety of training data. Although we could show that this is a rather unusual scenario, the approach may not be cost-effective for enterprise application usage profiles that show an intensive use of non-standard transaction types. For these customized components, white-box approaches need to be employed to develop individual performance models. Hence, performance predictions may result in conflicting decision support for standard and customized components that need to be aligned before alternative designs can be investigated. As part of future research, we embed this activity into an additional optimization problem. Furthermore, we plan to test and evaluate the effectiveness of additional prediction model types for different standard components including task types, e.g., batch jobs or dialog jobs of particular complexity levels. This approach may allow for the prediction of service performance on a higher level of granularity. Finally, we demonstrated performance predictions in order to verify service design alternatives that have been identified by optimization algorithms. This way, we were able to perform an end-to-end server consolidation process that optimizes the result with regard to both costs and service response times. Nevertheless, we are eager to further investigate the integration of our developed prediction models into the fitness function of a genetic algorithm and measure the effects on

solution quality and computing time. At present, together with our industry partner, we work on the automation of the described process. Therefore, a web interface is being developed to enable landscape optimizations and service predictions on demand. Hence, new business models in the domain of capacity management for enterprise applications are being enabled and may be subject to future research.

References

[1] L. Grinshpan, *Solving enterprise applications performance puzzles: queuing models to the rescue*. John Wiley & Sons, 2012. Available: https://doi.org/10.1002/9781118161920

[2] A. Beloglazov and R. Buyya, "Energy Efficient Resource Management in Virtualized Cloud Data Centers," in *10th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing*, 2010. Available: https://doi.org/10.1109/CCGRID.2010.46

[3] E. Tudenhöfner, *Integration of Performance Management into the Application Lifecycle*. diplom.de, 2011.

[4] G. Britain, V. Lloyd, D. Wheeldon, S. Lacy, and A. Hanna, *ITIL Continual Service Improvement*. Tso, 2011.

[5] Gartner, "Worldwide IT Spending Forecast." 2017 [Online]. Available: http://www.gartner.com/newsroom/id/3568917

[6] H. Müller, S. Bosse, M. Wirth, and K. Turowski, "Collaborative Software Performance Engineering for Enterprise Applications," in *Proceedings of the 50th Hawaii International Conference on System Sciences*, 2017. Available: https://doi.org/10.24251/HICSS.2017.047

[7] D. Filani, J. He, S. Gao, M. Rajappa, A. Kumar, P. Shah, and R. Nagappan, "Dynamic Data Center Power Management: Trends, Issues, and Solutions.," *Intel Technology Journal*, vol. 12, no. 1, 2008. Available: https://doi.org/10.1535/itj.1201.06

[8] A.-C. Orgerie, M. D. De Assuncao, and L. Lefevre, "A Survey on Techniques for Improving the Energy Efficiency of Large Scale Distributed Systems," *ACM Computing Surveys*, vol. 46, no. 4, pp. 1–35, Dec. 2014. Available: https://doi.org/10.1145/2532637

[9] H. Müller, S. Bosse, and K. Turowski, "Optimizing server consolidation for enterprise application service providers," in *Proceedings of the 2016 Pacific Asia Conference on Information Systems*, 2016.

[10] A. Beloglazov and R. Buyya, "Adaptive Threshold-Based Approach for Energy-Efficient Consolidation of Virtual Machines in Cloud Data Centers," in *Proceedings of the 8th International Workshop on Middleware for Grids, Clouds and e-Science*, 2010. Available: https://doi.org/10.1145/1890799.1890803

[11] A. Brunnert and H. Krcmar, "Continuous performance evaluation and capacity planning using resource profiles for enterprise applications," *Journal of Systems and Software*, vol. 123, pp. 239–262, 2015. Available: https://doi.org/10.1016/j.jss.2015.08.030

[12] J. Humble and D. Farley, *Continuous Delivery: Reliable Software Releases through Build, Test, and Deployment Automation (Adobe Reader)*. Pearson Education, 2010.

[13] A. Brunnert, A. van Hoorn, F. Willnecker, A. Danciu, W. Hasselbring, C. Heger, N. Herbst, P. Jamshidi, R. Jung, J. von Kistowski, and others, "Performance-oriented DevOps: A Research Agenda," *arXiv preprint arXiv:1508.04752*, 2015.

[14] B. Speitkamp and M. Bichler, "A Mathematical Programming Approach for Server Consolidation Problems in Virtualized Data Centers," *IEEE Transactions on Services Computing*, vol. 3, pp. 266–278, May 2010. Available: https://doi.org/10.1109/TSC.2010.25

[15] A. Shehabi, S. Smith, D. Sartor, R. Brown, M. Herrlin, J. Koomey, E. Masanet, N. Horner, I. Azevedo, and W. Lintner, "United States data center energy usage report," 2016.

[16] J. Koomey and J. Taylor, "New data supports finding that 30 percent of servers are 'Comatose', indicating that nearly a third of capital in enterprise data centers is wasted," *June*, vol. 3, p. 2015, 2015.

[17] J. M. Kaplan, W. Forrest, and N. Kindler, "Revolutionizing data center energy efficiency," Technical report, McKinsey & Company, 2008.

[18] H. Müller and S. Bosse, "Multidimensional Workload Consolidation for Enterprise Application Service Providers," in *Proceedings of the 26nd Americas Conference on Information Systems*, 2016.

[19] M. Stillwell, D. Schanzenbach, F. Vivien, and H. Casanova, “Resource Allocation Algorithms for Virtualized Service Hosting Platforms,” *Journal of Parallel and Distributed Computing*, vol. 70, pp. 962–974, Sep. 2010. Available: https://doi.org/10.1016/j.jpdc.2010.05.006

[20]SAP, “Measuring in SAPS.” 2017 [Online]. Available: https://www.sap.com/solution/benchmark/measuring.html

[21] J. Rolia, A. Andrzejak, and M. Arlitt, “Automating Enterprise Application Placement in Resource Utilities,” in *14th IFIP/IEEE International Workshop on Distributed Systems: Operations and Management (DSOM)*, 2003, vol. 2867, pp. 118–129. Available: https://doi.org/10.1007/978-3-540-39671-0_11

[22] J. Rolia, L. Cherkasova, M. Arlitt, and A. Andrzejak, “A capacity management service for resource pools,” in *5th International Workshop on Software and Performance (WOSP)*, 2005, pp. 229–237. Available: https://doi.org/10.1145/1071021.1071047

[23] M. Bichler, T. Setzer, and B. Speitkamp, “Capacity planning for virtualized servers,” in *Workshop on Information Technologies and Systems (WITS)*, 2006.

[24] D. A. Menascé and P. Ngo, “Understanding Cloud Computing: Experimentation and Capacity Planning.,” in *Int. CMG Conference*, 2009.

[25] S. Becker, H. Koziolek, and R. Reussner, “The Palladio component model for model-driven performance prediction,” *Journal of Systems and Software*, vol. 82, no. 1, pp. 3–22, 2009. Available: https://doi.org/10.1016/j.jss.2008.03.066

[26] F. Brosig, N. Huber, and S. Kounev, “Architecture-level software performance abstractions for online performance prediction,” *Science of Computer Programming*, vol. 90, pp. 71–92, 2014. Available: https://doi.org/10.1016/j.scico.2013.06.004

[27] D. Westermann, J. Happe, M. Hauck, and C. Heupel, “The performance cockpit approach: A framework for systematic performance evaluations,” in *Software Engineering and Advanced Applications (SEAA), 2010 36th EUROMICRO Conference on*, 2010, pp. 31–38. Available: https://doi.org/10.1109/SEAA.2010.24

[28] S. Balsamo, A. D. Marco, P. Inverardi, and M. Simeoni, “Model-based performance prediction in software development: A survey,” *IEEE Transactions on Software Engineering*, vol. 30, no. 5, pp. 295–310, 2004. Available: https://doi.org/10.1109/TSE.2004.9

[29] D. A. Menasce, “Composing web services: A QoS view,” *Internet Computing, IEEE*, vol. 8, no. 6, pp. 88–90, 2004. Available: https://doi.org/10.1109/MIC.2004.57

[30] C. U. Smith, “Increasing Information Systems Productivity by Software Performance Engineering,” in *Seventh International Computer Measurement Group Conference, New Orleans, LA, USA, December 1–4, 1981, Proceedings*, 1981, pp. 5–14 [Online]. Available: http://www.cmg.org/?s2member_file_download=/proceedings/1981/81INT002.pdf

[31] D. Tertilt and H. Krcmar, “Generic performance prediction for ERP and SOA applications.,” in *ECIS*, 2011.

[32] N. Pollock, R. Williams, and R. Procter, “Fitting standard software packages to non-standard organizations: the ‘biography’of an enterprise-wide system,” *Technology Analysis & Strategic Management*, vol. 15, no. 3, pp. 317–332, 2003. Available: https://doi.org/10.1080/0953732031000160150 4

[33] T. M. Somers and K. Nelson, “The impact of critical success factors across the stages of enterprise resource planning implementations,” in *System Sciences, 2001. Proceedings of the 34th Annual Hawaii International Conference on*, 2001, p. 10–pp. Available: https://doi.org/10.1109/HICSS.2001.927129

[34] T. Rabl, S. Gómez-Villamor, M. Sadoghi, V. Muntés-Mulero, H.-A. Jacobsen, and S. Mankovskii, “Solving big data challenges for enterprise application performance management,” *Proceedings of the VLDB Endowment*, vol. 5, no. 12, pp. 1724–1735, 2012. Available: https://doi.org/10.14778/2367502.2367512

[35] M. J. Sydor, K. Sleeth, J. Toigo, E. Yourdon, S. E. Donaldson, S. G. Siegel, and G. Donaldson, *Apm best practices: Realizing application performance management*. Apress, 2010. Available: https://doi.org/10.1007/978-1-4302-3142-4

[36] Gartner, “Gartner IT Glossary.” 2017 [Online]. Available: http://www.gartner.com/it-glossary/application-performance-monitoring-apm

[37] V. A. Almeida, “Capacity planning for web services techniques and methodology,” in *IFIP International Symposium on Computer Performance Modeling, Measurement and Evaluation*, 2002, pp. 142–157. Available: https://doi.org/10.1007/3-540-45798-4_7

[38] H. Müller, S. Bosse, M. Pohl, and K. Turowski, “Capacity Planning as a Service for Enterprise Standard Software,” in *Business Informatics (CBI), 2017 IEEE 19th Conference on*, 2017, vol. 1, pp. 167–175. Available: https://doi.org/10.1109/CBI.2017.25

[39] A. R. Hevner, S. T. March, J. Park, and S. Ram, “Design Science in Information Systems Research,” *Management Information Systems Quarterly*, vol. 28, no. 1, pp. 75–105, May 2004. Available: https://doi.org/10.2307/25148625

[40] K. Peffers, T. Tuunanen, M. A. Rothenberger, and S. Chatterjee, “A design science research methodology for information systems research,” *Journal of Management Information Systems*, vol. 24, no. 3, pp. 45–77, 2007. Available: https://doi.org/10.2753/MIS0742-1222240302

[41] L. Cherkasova and J. Rolia, “R-Opus: A composite framework for application performability and QoS in shared resource pools,” in *International Conference on Dependable Systems and Networks*, 2006. Available: https://doi.org/10.1109/DSN.2006.59

[42] T. Setzer and A. Stage, “Decision support for virtual machine reassignments in enterprise data centers,” in *IEEE/IFIP Network Operations and Management Symposium (NOMS) Workshops*, 2010. Available: https://doi.org/10.1109/NOMSW.2010.5486597

[43] E. Feller, L. Rilling, and C. Morin, “Energy-aware ant colony based workload placement in clouds,” in *Proceedings of the 2011 IEEE/ACM 12th International Conference on Grid Computing.*, 2011. Available: https://doi.org/10.1109/Grid.2011.13

[44] H. A. Simon, “Administrative behavior; a study of decision-making processes in administrative organization-3,” 1976.

[45] N. Milanovic and B. Milic, “Automatic Generation of Service Availability Models,” *IEEE Transactions on Service Computing*, vol. 4, no. 1, pp. 56–69, 2011. Available: https://doi.org/10.1109/TSC.2010.11

[46] A. Immonen and E. Niemelä, “Survey of reliability and availability prediction methods from the viewpoint of software architecture,” *Software & Systems Modeling*, vol. 7, no. 1, p. 49, 2008. Available: https://doi.org/10.1007/s10270-006-0040-x

[47] S. Bosse, “Predicting an IT Service’s Availability with Respect to Operator Errors,” in *Proceedings of the 19th Americas Conference on Information Systems (AMCIS)*, 2013.

[48] Q. Zhang, L. Cherkasova, N. Mi, and E. Smirni, “A regression-based analytic model for capacity planning of multi-tier applications,” *Cluster Computing*, vol. 11, no. 3, pp. 197–211, 2008. Available: https://doi.org/10.1007/s10586-008-0052-0

[49] Gartner, “Magic Quadrant for Application Performance Monitoring Suites.” 2017 [Online]. Available: https://www.gartner.com/doc/reprints?id=1

[50] Y. Dittrich, S. Vaucouleur, and S. Giff, “ERP customization as software engineering: knowledge sharing and cooperation,” *IEEE software*, vol. 26, no. 6, 2009. Available: https://doi.org/10.1109/MS.2009.173

[51] V. A. Almeida and D. A. Menascé, “Capacity planning an essential tool for managing Web services,” *IT professional*, vol. 4, no. 4, pp. 33–38, 2002. Available: https://doi.org/10.1109/MITP.2002.1046642

[52] U. Fayyad, G. Piatetsky-Shapiro, and P. Smyth, “From data mining to knowledge discovery in databases,” *AI magazine*, vol. 17, no. 3, p. 37, 1996.

[53] Fujitsu, “Software Optimization Services - SAP SystemInspection Service.” 2016 [Online]. Available: http://sp.ts.fujitsu.com/dmsp/Publications/public/db-services-optimization-services-sap-systeminspection-de.pdf

[54] J. I. Maletic and A. Marcus, “Data cleansing: A prelude to knowledge discovery,” in *Data Mining and Knowledge Discovery Handbook*, Springer, 2009, pp. 19–32. Available: https://doi.org/10.1007/978-0-387-09823-4_2

[55] A. H. Fielding and J. F. Bell, “A review of methods for the assessment of prediction errors in conservation presence/absence models,” *Environmental conservation*, vol. 24, no. 01, pp. 38–49, 1997. Available: https://doi.org/10.1017/S0376892997000088

[56] M. R. Berthold, C. Borgelt, F. Höppner, and F. Klawonn, *Guide to intelligent data analysis: how to intelligently make sense of real data*. Springer Science & Business Media, 2010. Available: https://doi.org/10.1007/978-1-84882-260-3

[57] D. Mayer and D. Butler, “Statistical validation,” *Ecological modelling*, vol. 68, no. 1–2, pp. 21–32, 1993. Available: https://doi.org/10.1016/0304-3800(93)90105-2

[58] D. L. Shaeffer, “A model evaluation methodology applicable to environmental assessment models,” *Ecological Modelling*, vol. 8, pp. 275–295, 1980. Available: https://doi.org/10.1016/0304-3800(80)90042-3

[59] D. S. Liu, K. C. Tan, S. Y. Huang, C. K. Goh, and W. K. Ho, “On solving multiobjective bin packing problems using evolutionary particle swarm optimization,” *European Journal of Operational Research*, vol. 190, pp. 357–382, Oct. 2008. Available: https://doi.org/10.1016/j.ejor.2007.06.032

[60] K. Wilhelm, “Capacity Planning for SAP-Concepts and tools for performance monitoring and modelling,” *CMG Journal of Computer Resource Management*, no. 104, 2001.

[61] G. Kirchgässner, J. Wolters, and U. Hassler, *Introduction to modern time series analysis*. Springer Science & Business Media, 2012. Available: https://doi.org/10.1007/978-3-642-33436-8

[62] H. Li, G. Casale, and T. Ellahi, “SLA-driven planning and optimization of enterprise applications,” in *Proceedings of the first joint WOSP/SIPEW international conference on Performance engineering*, 2010, pp. 117–128. Available: https://doi.org/10.1145/1712605.1712625

[63] S. Bosse, M. Splieth, and K. Turowski, “Multi-objective optimization of IT service availability and costs,” *Reliability Engineering & System Safety*, vol. 147, pp. 142–155, 2016. Available: https://doi.org/10.1016/j.ress.2015.11.004

[64] T. Gneiting, “Making and evaluating point forecasts,” *Journal of the American Statistical Association*, vol. 106, no. 494, pp. 746–762, 2011. Available: https://doi.org/10.1198/jasa.2011.r10138

[65] K. Deb, S. Agrawal, A. Pratap, and T. Meyarivan, “A Fast Elitist Non-dominated Sorting Genetic Algorithm for Multi-objective Optimization: NSGA-II,” in *Proceedings of the 6th International Conference on Parallel Problem Solving from Nature*, 2000, vol. 1917. Available: https://doi.org/10.1007/3-540-45356-3_83

[66] J. Xu and J. A. B. Fortes, “Multi-objective Virtual Machine Placement in Virtualized Data Center Environments,” in *IEEE/ACM International Conference on Cyber, Physical and Social Computing (CPSCom)*, 2010. Available: https://doi.org/10.1109/GreenCom-CPSCom.2010.137

[67] Y. Gao, H. Guan, Z. Qi, Y. Hou, and L. Liu, “A multi-objective ant colony system algorithm for virtual machine placement in cloud computing,” *Journal of Computer and System Sciences*, vol. 79, pp. 1230–1242, Dec. 2013. Available: https://doi.org/10.1016/j.jcss.2013.02.004