Association for Information Systems
AIS Electronic Library (AISeL)

Wirtschaftsinformatik Proceedings 2007 Wirtschaftsinformatik

February 2007

Services and resource profiles as metrics for the allocation of IT infrastructure costs

Reinhard Brandl
Technische Universität München, brandlr@in.tum.de

In: Oberweis, Andreas, u.a. (Hg.) 2007. *eOrganisation: Service-, Prozess-, Market-Engineering*; 8. Internationale Tagung Wirtschaftsinformatik 2007. Karlsruhe: Universitätsverlag Karlsruhe

ISBN: 978-3-86644-094-4 (Band 1)
ISBN: 978-3-86644-095-1 (Band 2)
ISBN: 978-3-86644-093-7 (set)

Services and resource profiles as metrics for the allocation of IT infrastructure costs

Reinhard Brandl

Chair of Internet-based Information Systems (IBIS)
Institute of Informatics
Technische Universität München
85748 Garching, Germany
brandlr@in.tum.de

Abstract

This paper addresses the general problem of allocating costs for shared and distributed IT infrastructures to business entities, like cost centres or processes. In contrast to earlier times, when monolithic mainframes were the central resources in data centres, client/server architectures with distributed heterogeneous components now dominate. In such an environment, resource consumption (e.g. CPU time) usually cannot be fully apportioned to users or other business entities. We therefore propose an approach which is based on predetermined resource profiles as estimates for the mean resource consumption of business services. Instead of metering every component during operation and consolidating the log data afterwards, only service invocations need to be logged. For the determination of the resource profiles we developed a measurement methodology and a software toolkit. The approach was successfully tested and evaluated for OLTP systems at the BMW Group. Furthermore, by combining resource profiles with Queuing Network Theory, we could demonstrate their appropriateness for capacity planning.

1 Introduction

Throughout the literature on IT management and controlling, three general topics are commonly found: firstly, the minimisation of operational and security risks; secondly, the increasing of IT Business Value (effectiveness); and thirdly, the need for a continuous reduction of costs (efficiency). In the context of the latter two aspects, one of the major challenges, particularly for internal IT managers, is to establish a transparent relationship between IT spending and business entities like cost centres, cost units, processes or activities. Otherwise, an IT unit runs the risk of appearing as black box with high fixed costs and no measurable business value. The lack of cost transparency of internal IT service provision is one of the major motivations for outsourcing activities [DGHJ04].

Allocating costs to business entities is a particularly intricate task when the corresponding resource is shared and an apportionment of its consumption is not practicable. In the IT context, this is usually the case for corporate data centres and infrastructure services. Whereas for desktop- and office-oriented services a known customer exists, the consumption of infrastructure resources usually cannot be fully apportioned to users or other business entities. As a single user activity might involve a multitude of separated components (e.g. servers, network, storage, etc.) which mostly have no access to the original business context of the operation, a consolidation of the distributed log information is elaborate and error-prone. As long as major resources are dedicated, e.g. one server per application, direct costing approaches can be applied. However, with the advances in virtualisation technologies and the move towards shared resource pools, new accounting concepts and metrics are required.

We address this problem in the context of distributed OLTP systems, which are the beating heart of modern enterprise computing environments. We assume first that an OLTP system provides one or more business services to its users and secondly that the invocation of such a service always results in similar resource consumption (e.g. transferred bytes, CPU seconds). The idea is to determine for each service a complete resource profile in a test environment. During regular operations, only service invocations per user or cost centre are logged. If the approach is practicable, the information would be sufficient for a usage-based cost apportionment of IT infrastructures. A distributed metering of resource consumption could be omitted. Furthermore, the resource profiles might be valuable inputs for capacity planning.

We tested the approach, under realistic conditions, in the central IT unit of the BMW Group. In a first step, we developed a methodology and a software tool for the determination of resource profiles in heterogeneous environments. In a second step of the project, we used these profiles for the parameterisation of Queuing Network Models (QNM). The results of these models were compared with the outcomes of real load tests. We had two main reasons for undertaking this procedure. Firstly, the resource profiles should be validated and, secondly, their appropriateness for capacity planning should be demonstrated. In this paper we present our results.

The remainder of the article is structured as followed. Chapter 2 introduces the problem of IT infrastructure cost allocation and briefly discusses current practiced approaches. In Chapter 3 we present our idea of allocating infrastructure costs via resource profiles. The concept is further illustrated in Chapter 4 by the example of the central IT unit of the BMW Group. The overall results are evaluated in Chapter 5. The paper concludes with an outlook on future areas of research in Chapter 6.

2 Problem statement

Cost accounting is a managerial accounting activity which increases transparency and supports decisions by allocating overhead and direct costs to business entities. Traditionally, business entities are cost centres (e.g. departments, subsidiaries) or cost units (e.g. product or service produced), but more recent forms of cost accounting also consider processes [HoMa89] or activities [KaBr87]. The goal is to measure the economic performance of business entities and the value of the resources consumed in producing goods and services [Harp93]. Cost transparency is a quite general need. However, the purposes the information is used for (e.g. internal chargeback, benchmarks or management evaluation), and to what extent, is strongly organisation-specific.

IT is mostly organised as a service function which supports more than one business entity. The budget is dominated by salaries, consulting fees and infrastructure costs. As a direct breakdown of those costs to business entities is for the most part not feasible, one can either treat them as general overhead or try to identify appropriate cost drivers as accounting objects (IT products or services). They should enable a usage-based cost allocation and serve as bridges between IT and business. On the IT side they integrate different resources and cost types; on the business side they allow usage and cost control. In the context of desktop- and office-oriented IT services, accounting objects could be the provision of a desktop PC, an e-Mail Box or a telephone line. For non-standardised IT services, the identification of accounting objects is more sophisticated. Consider the management and operation of business applications, hosted in data centres. In the example in Figure 1, there is one OLTP system used by five different business functions.

From their point of view, the provision and/or usage of the system and the related support services are reasonable accounting objects. As the percentage rates indicate, a good portion of the total costs can be allocated either to the application or directly to the business functions. We assume that ratios (e.g. number of logins, number of employees, transactions) for an apportionment of the application-specific costs (development, maintenance, etc.) exist and focus in the following on the allocation of the remaining infrastructure costs.

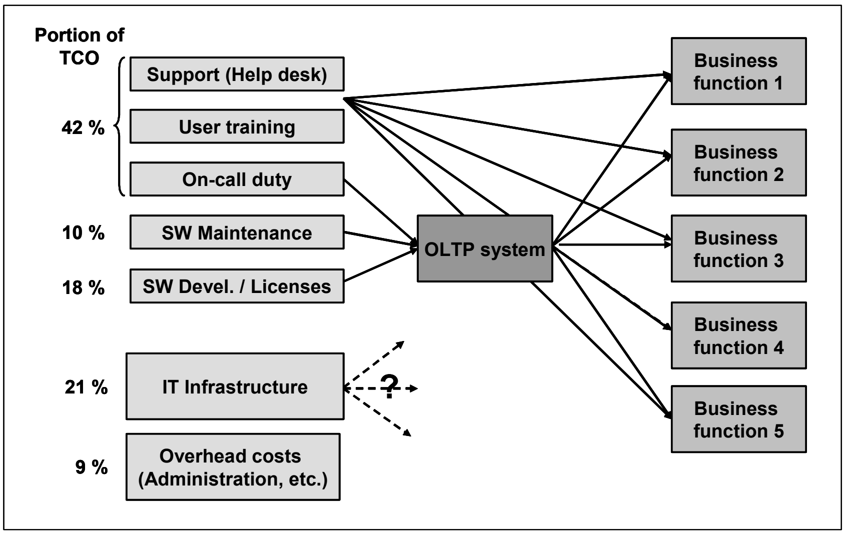

Figure 1: Costs of an OLTP system. (Rates adapted from [Zieh04])

In the literature (e.g. [Aure97; Bert01; Hein02]), the ITIL library [OoOC01], and in our cooperation with the BMW Group (Chapter 4), we identified several ways how organisations handle these costs.

Direct cost allocation. In case a resource (e.g. a server) is dedicated to a specific application or a business unit, the incurring costs can be directly allocated. This proceeding is transparent and easy to implement. However, this approach is mainly limited to computing infrastructure. Other resources, like network or EAI components, are usually shared. With regard to the average server utilisation of less than 25% in modern data centres and the cost saving potential of virtualisation technologies, dedicated resources will more and more give way to shared resource pools.

Overhead costs. Infrastructure costs which cannot be directly allocated are handled in analogy to other IT overhead costs (administration, buildings, etc.). Particularly for small to medium sized companies, with IT organised as a service cost centre, this approach might be sufficient.

Measured usage. At first glance, measuring resource consumption and allocating it to the originator is the most obvious method for the apportionment of shared infrastructures costs. A measured usage approach is reasonable for certain resources like disk space or telephony as well as for very computing-intensive jobs, like batch processing or simulations. However, in an OLTP context, measuring and apportioning resource consumption to a specific user or even to an application is not trivial. In contrast to earlier times, when monolithic mainframes were the dominating resources, modern OLTP systems are based on the client/server model and integrate different and distributed components. A single business transaction may cause resource consumptions on multiple components. In a simple three-tier architecture, with a web server, application server and database server, three different computing resources are already involved. Due to performance and security reasons (and the lack of standards), the original business context of a transaction is mostly not available in the backend. For instance, user names are not further transmitted after a successful authorization and for database access connection pools are used. This lack of information complicates the establishment of an end-to-end perspective. So, particularly in heterogeneous environments, the collection and consolidation of log data for accounting purposes quickly becomes tedious.

Per provided application. The provision of an application in a data centre is an "IT product" to which cost are allocated, either by a single or a tiered flat rate (e.g. classification according to the expected number of concurrent users, required service levels or number of interfaces). The BMW Group uses such flat rates for applications on its J2EE infrastructure. This approach is particularly easy to implement, as no explicit differentiation of resource costs and no metering is required. The accompanying lack of transparency may be accepted, if the applications are roughly of a similar nature (complexity, resource consumption, etc.). However, the notion of an "application" is a little bit fuzzy. Depending on the software architecture, the same business functionality may be implemented in a single or in multiple separate applications (i.e. executables, ear-files, etc.). Furthermore, one user of an application can cause a lot more resource consumption than 100 users of another. If cost accounting is combined with chargeback, this simplification potentially leads to acceptance problems.

In our opinion, none of the approaches presented above really cope with the complexity of modern OLTP systems. It's not surprising that the lack of cost transparency is one of the major reasons for outsourcing IT activities. In a market environment, accounting metrics can be found which are not directly linked to costs (e.g. "pay-per-business transaction"), but which rely on a risk sharing model between customer and service provider. This establishes cost transparency at the customer side, but does not solve the fundamental problem.

3 Resource profiles for accounting and capacity planning

In search of alternatives, we abstracted from the technical view on “applications” and took a business perspective, with an IT system offering a number of useful services (e.g. „process order“, „update stock“, and „add customer“). An application may implement one or more of those services, but a service can also comprise multiple applications. The invocation of a service causes resource consumption in the infrastructure (CPU time, transferred bytes, etc.). The exact amount is dependent on input factors like parameters passed or the internal states of the components involved. Nevertheless, for repeated service invocations an average value or, more precisely, a vector with average consumption values for different resources, exists. If these vectors and the number of service invocations per user were known, a consumption-based cost apportionment to services and users would be possible. The elaborate process of measuring and consolidating log data from different components could be bypassed. Furthermore, these vectors would be valuable inputs for the alignment of business forecasting and IT capacity planning.

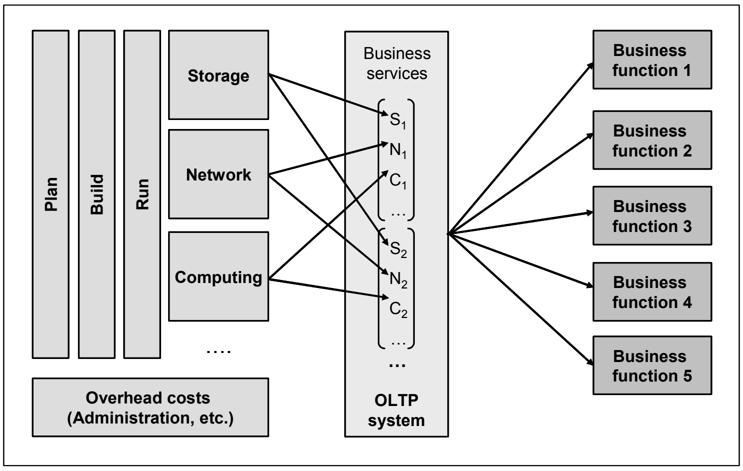

Figure 2: Cost allocation by business services and resource profiles.

Whether such an approach is actually practicable in an industrial context depends on the efforts required for the determination of appropriate services and average consumption vectors. As the problem of allocating infrastructure costs is of a quite general nature, we further followed this idea and analysed its potential in cooperation with our industrial partner, the BMW Group.

4 Results and experiences from the pilot test

4.1 Organisational context

At the BMW Group, management and operation of the considered OLTP systems is organised on two levels. A central IT department provides standardised infrastructure services, such as server computing, network and storage for the whole company. At each Business Unit (e.g.

Production, Sales) a dedicated IT department is responsible for the management and operation of their applications. The Business Units are charged for their consumption of infrastructure resources. If resources are dedicated, direct costing is applied. Dedicated resources are required in particular for business-critical applications with peak demands. However, most applications run on shared infrastructure and we assume that with increasing hardware performance and advances in virtualisation technologies, their share will further grow.

The predominant application platform at the BMW Group is Java/J2EE. The central IT department therefore provides standardised environments on shared and dedicated computers. The Business Units are either charged per application (shared infrastructure) or by the performance characteristics of their dedicated computers. These flat rates cover all other resource costs. This approach is quite efficient and comprehensible. However, it does not consider the different nature of applications and their resource consumption (see Chapter 2). In a pilot test, we evaluated if the approach, presented in the previous chapter, would be a feasible alternative.

4.2 Integration of the approach into existing IT processes

The IT processes of the BMW Group specify that prior to the first deployment or after a major software change, every application has to pass an operational approval. This process consists of several well-defined tests (e.g. installation, load and stress tests, cluster failover) in an isolated environment. In contrast to tests during the development phase, the operational approval is conducted by infrastructure specialists and not by the responsible software engineers. The objective is to verify whether the application meets stability and performance requirements (at the BMW Group also referred to as “operational maturity”). Such approval or acceptance tests are common practice in industrial data centres [OoOC02] and we consider them as a suitable opportunity for the determination of the required resource profiles. So, we designed an additional process step for the operational approval at the BMW Group, to acquire consumption data of typical user behaviour.

4.3 A methodology for the determination of resource profiles

As relevant run-time consumption metrics, we identified CPU time, network and storage I/O. We excluded disk space, as it is usually a priori allocated to a specific application or a database and elaborate accounting tools are available. Memory is usually also considered as a scarce resource. However, the maximum amount of physical memory a server can allocate on a machine is already determined at startup (e.g. by setting a range for virtual memory) and we propose to take this value as the basis for accounting.

The determination of resource profiles in heterogeneous environments is not trivial. Existing profiling tools focus on diagnosing performance problems, like memory leaks or CPU consumption of methods or transactions. They are technology- or vendor-specific, (e.g. Java/J2EE [JaPeoJ], .NET [Schw06], Intel [InteloJ], different ERP/CRM systems [SymaoJ]) and do not cover all the resources mentioned above. Load generators (e.g. Mercury Loadrunner [MercoJ], SilkPerformer [SeguoJ], The Grinder [TheGoJ], httperf [MoJi98]), on the other hand, have elaborate means to simulate different user behaviours in various environments. However, their main analysis focus lies on response times experienced by their virtual clients. Although most tools provide consoles for a remote monitoring of hardware and server performance, none of the products we evaluated calculates resource consumption for specific user activities.

To overcome the limitations of existing tools, we developed a new methodology and an appropriate software toolkit. The underlying motivation was to enable a flexible and non-invasive determination of resource profiles for business-oriented services, independent from underlying hardware and software technologies. The basic idea is to install the system under consideration in an isolated environment (as is usual for operational approvals), simulate consecutive service invocations by means of a load generator and correlate start and end times of services with performance log files recorded at the different components. The prerequisite of the approach is that no other users or applications distort the measurements. So, besides inevitable background activities, resource consumption can be fully allocated to service invocations. The corresponding software toolkit consists of three separate packages. Firstly, a collection of log file parsers to transfer the measurements into a database, secondly, an analysis component which correlates the data and computes the resource profiles, and finally, a tool for the visualisation of the results.

4.4 Experimental setup

We tested the approach with several applications. For the experiments we set up an infrastructure which is similar to the standards of the BMW Group (see Figure 3). To increase the empirical coverage, we combined different Operating Systems (Linux, Windows and Unix) and servers (Apache HTTP, Bea Weblogic and Oracle Database).

As example scenario for this publication we use the J2EE reference implementation Java PetStore [SunoJ]. We chose this example for three major reasons. Firstly, it is publicly available and well documented, secondly, as reference implementation, it covers most J2EE technologies (EJBs, Servlets, JSPs, Web Services and XML, JMS, CMP, etc.), and thirdly, the software architecture, with several interacting applications in the front- and back-end, is an appropriate representation of the structure of modern enterprise systems. For the simulation of service invocations, we chose Mercury Loadrunner [MercoJ], as it had been already used during the operational approval at the BMW Group. To monitor the resource consumption at the different computers, we relied on standard OS tools (Unix/Linux: sar, Winodws: perfmon). Thus, we were independent from server technologies and could avoid the installation of additional software.

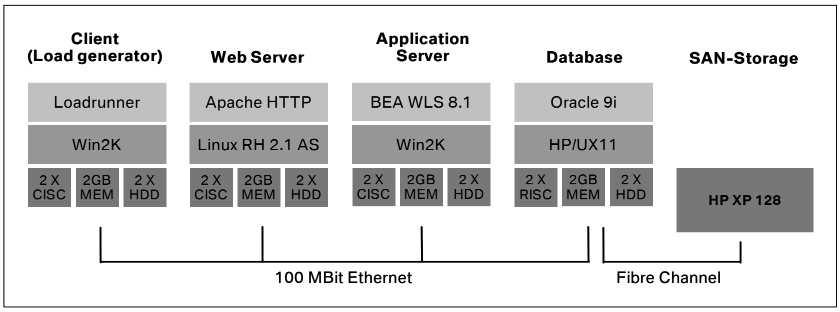

Figure 3: Infrastructure used in the experiments

4.5 Definition of services

We assume that, from a business perspective, an IT system provides a number of useful services and that we can determine resource profiles as estimates for the mean resource consumption of service invocations. To minimise the gap between estimated and actual resource consumption one could define every possible user activity within the system as a separate service. Nevertheless, in our opinion a large number of fine-granular services would lead to an unreasonable complexity for cost accounting. We propose instead to consider whole or at least parts of business processes as services. For instance, in the PetStore case, we defined the shopping process of an average user as single service. The logging of service invocations could be easily combined with central authentication/authorisation mechanisms. However, as visitors have different possibilities of using the store, i.e. the service, information about the average user behaviour must be available or estimated. This might be a problem, in particular for new systems. In that case, we recommend reviewing the assumptions on services and usage after an observation period. For the operational approval at the BMW Group, software projects were already obligated to specify the expected usage and provide appropriate load test scripts for the different use cases. We proposed to take this information as the basis for the definition of services.

4.6 Example: Determination of the resource profile for a PetStore shopper

For the determination of user paths through web applications, several tools (e.g. WebTrends Analytics from WebTrend Inc. [WebToJ]) and scientific approaches exist (e.g. Customer Behaviour Model from Menascé and Almeida [MeAl00]) exist. For the PetStore example, we assume that such information is available and we recorded the path of an average shopper in a load test script. As mentioned above, we use standard OS tools for the performance monitoring at the different computers. The major drawback of those tools is that they only provide rough measurements, which are furthermore distorted by background activities. For instance, the minimal interval between two data points is one second, which is much higher than the usual response time of a three-tier web application, like the Java PetStore. Accordingly, the resource consumption of a single PetStore shopper would be hardly measurable. To solve this problem, and to separate the resource consumption of services from background activities, we

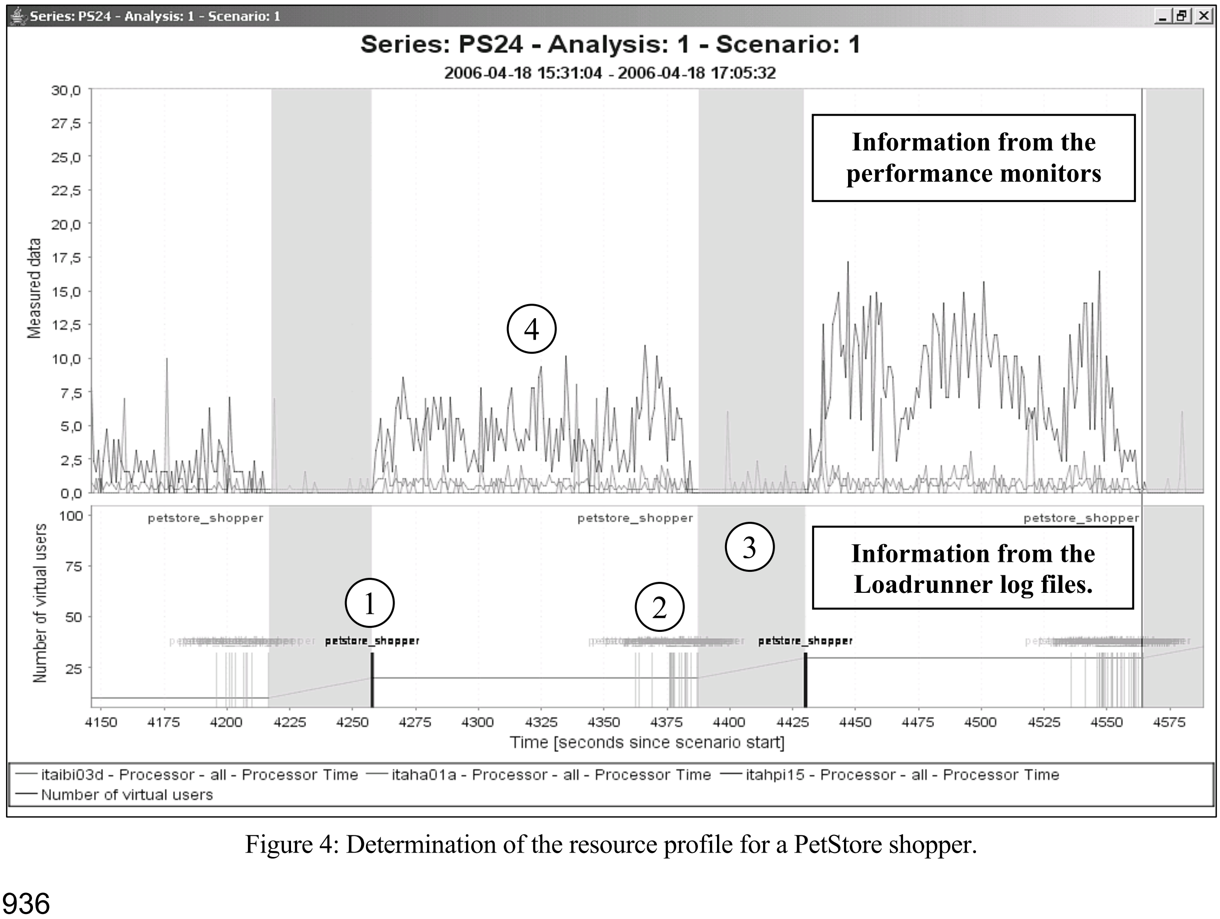

Figure 4: Determination of the resource profile for a PetStore shopper.

successively raise the number of parallel service invocations and determine by means of a linear regression the increase of resource consumption. In the PetStore example, we started with 10 concurrent users and increased the number in steps of 10 until we reached 70 users and repeated this process several times. Figure 4 illustrates this concept with a screenshot from our analysis tool. At label (1), 20 concurrent users start browsing through the PetStore. To avoid peaks, we build in arbitrary think times between the requests. About two minutes later all users have finished their shopping tour (2). After an idle phase of 45 seconds (3), the procedure is restarted with 30 concurrent users. During the whole test, the performance monitors at the Web, Application and Database Servers record the system behaviour in log files on their local disks. The upper diagram (4) shows for instance the overall CPU utilisation ((user + system time) / 2 processors) at the three different computers. The most utilised component is the Application Server. Data from network and storage resources is analysed in the same manner.

After the test, log file parsers consolidate the data and copy it into a database. Then the analysing component correlates start and end times of transactions with selected performance data and computes the average resource consumption for every measurement cycle. In the following, we apply a linear regression to these values and compute the resource consumption per service invocation (i.e. slope of the regression line). We use the correlation coefficient (r є [-1;1]) as a first quality indicator for the results. Figure 5 shows the results of the analysis of CPU times. Accordingly, a single PetStore shopper consumes during a 2 minute visit 0.207 seconds of CPU time[1] at the Application Server (r=0.825), 0.012 seconds at the Web Server (r=0.968) and 0.025 seconds at the Database (r=0.875). For the other resources in the profile – network (transferred bytes) and storage (transferred blocks) – we got similar results. Nevertheless, the coefficient of correlation only evaluates the fitting of the regression line and not the resource consumption as such. So, we were looking out for a further sanity check for the resource profiles.

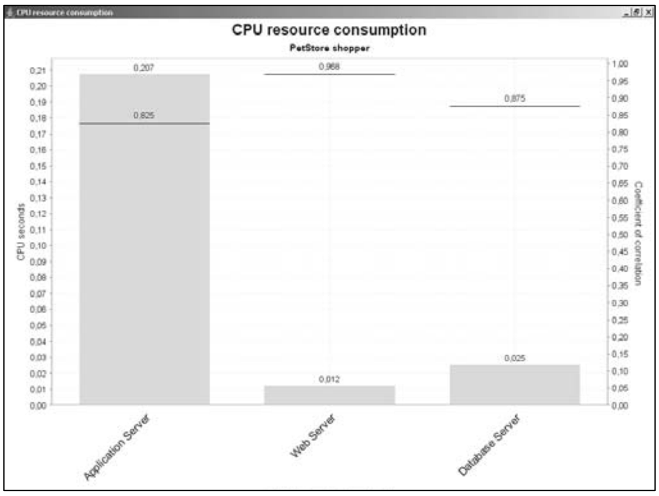

Figure 5: CPU resource consumption of a PetStore shopper

[1] These are raw values, dependent on the architecture and performance characteristics of the hardware used during the tests. For accounting purposes, they must be normalised. We therefore propose the usage of standard industry benchmarks.

4.7 Resource profiles as input parameters for analytical capacity planning

Queuing Network Theory is a well-studied analytical approach for performance modelling and capacity planning of distributed computer systems. However, is rarely used in practice. One reason for this is that the input parameters, like service time for jobs, are not directly available from common performance monitors. The resource profiles presented above might fill this gap. We further evaluated this idea, for two major reasons. By comparing performance predictions with measured values, we could, firstly, validate the resource profiles (see previous section) and, secondly, test their appropriateness for capacity planning. In the following section we briefly present the results.

As the first step, we developed an appropriate load test set-up. We use the same scripts as for the determination of the resource profiles. However, instead of simultaneous starts and stops, we put the users in an endless loop. Every five minutes, we added ten more users until the first component reaches its bottleneck. During the load test, performance monitors record the system behaviour in log files. The data is afterwards copied into a database, but instead of analysing resource consumption, we determine average values for utilisation and response times.

In a second step, we developed a general performance model for the three-tier infrastructure used during the experiments (see Figure 3). We separately analysed each of the above-mentioned five minute slots. As the number of users remains constant within each slot, we applied a Closed Queuing Network Model (see Figure 6). As the solution algorithm, we implemented the Mean Value Analysis algorithm [ReLa80]. For further information on Queuing Network Theory, we refer the interested reader to the work of Menascé *et. al.* [MeAl00; MeAD04] and Bolch *et al.* [BoGM06].

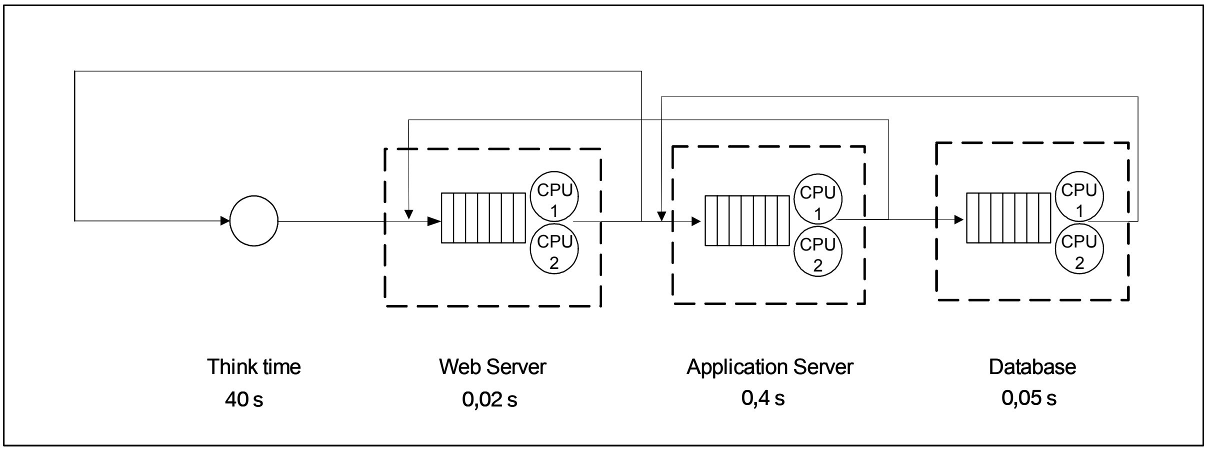

Figure 6: QN-Model of the infrastructure used in the experiments.

In the PetStore case we solely modelled the processors, as the time required for disk accesses was negligible. Due to the dual-processor architecture of the considered hardware (see Figure 3), we modelled each server as a queue with two separate stations. The service demands were taken from the resource profiles. As the values in Figure 5 are based on the average CPU utilisation of both processors, we had to double them for the performance model.

We conducted a load test according to the above-mentioned process and compared the outcome to the predictions of the Queuing Network model. As the length of the think time between two requests must not have an effect on the resource consumption, we intentionally changed it for this load test (in total 40 seconds instead of 120 seconds). The bottleneck in the PetStore example was the Application Server. Figure 7 compares its measured utilisation (average value of the time interval with constant number of users) to the computed values from the Queuing Network Model. The maximum absolute deviation is about 10%, which we consider to be tolerable. Nevertheless, we are still looking for improvements.

Analytical capacity planning has a number of advantages over common approaches. Once a validated model exists, the effect of parameter changes, concerning the hardware performance, the expected load and the user behaviour can be evaluated. These "what-if"-analyses are particularly helpful in situations where the hardware from the test environment differs from the production environment. The results are usually more precise and substantiated than "rule of thumb" estimates or general sizing guidelines from hard- and software vendors.

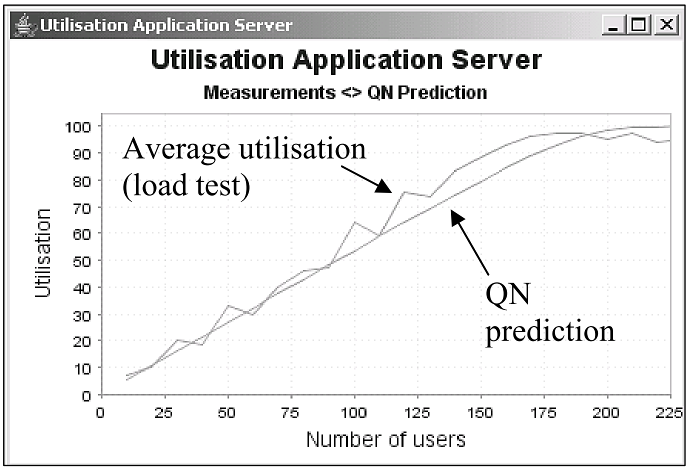

Figure 7: Comparison of average CPU utilisation during a load test with QN predictions.

4.8 Contribution and related approaches

The contribution of our concept for the determination of resource profiles is threefold. Firstly, by utilising the capabilities of a professional load generator, we can determine resource profiles not only for single transactions, but for whole user sessions or business processes. Until now we tested it only for applications with web-based interfaces. However, this should not be a restriction. Tools like the Mercury Loadrunner can simulate load in an analogous manner on a multitude of different front- and backend interfaces. Secondly, as we fully rely on standard performance monitors of Linux/Unix and Windows Operating Systems, new components can be easily integrated as the case arises. Despite the limitations of these tools (e.g. minimal measurement intervals of one second), we could demonstrate that the concept is also appropriate for very small values of resource consumption. Thirdly, through the combination of Queuing Network Theory and load tests we provide means for a quick sanity check of the resource profiles.

As illustrated in the Section 4.3, the determination of overall resource profiles for services on distributed systems is not yet supported by tools available on the software market. This problem had been also addressed by Nagaprabhanjan and Apte who recently presented a tool [NaAp05] for automated resource consumption profiling of distributed transactions. Analogous to our approach, they combine load generation with OS performance monitors for the determination of resource profiles. However, their focus is not accounting, but the determination of input parameters for performance analysis and capacity planning. The major strength of Nagaprabhanjan and Apte is the run-time coordination of load generation and resource profiling. This enables more precise measurements. Accordingly, they require fewer measurement cycles and generate less data for the analysis. Our approach is more flexible concerning the definition of services and the integration of additional resources. However, due to the above mentioned aspects it takes more time for measurement and analysis.

5 Evaluation of results

We could demonstrate during the project that it is technically feasible to determine comprehensive resource profile for business-oriented services in a distributed and heterogeneous environment. However, our methodology is subject to the following two major conditions. Firstly, we assume that knowledge about appropriate services and user behaviour is available. This assumption may not hold for new applications. If no historical usage data exists, measurements have to be based on estimations. This can lead to distorted resource profiles. Secondly, our metering concept requires an environment with no external interferences. In organisations with formal IT testing and approval processes, such an infrastructure might be given. If not, considerable extra efforts and investments are necessary. In that case the efficiency of the approach has to be seriously questioned.

If the above requirements are fulfilled and if there is a need for usage-based cost allocation, the approach is a feasible alternative to “measured usage”. Beyond its operational advantages (a distributed metering of resource consumption can be omitted) and its business orientation (service invocation instead of CPU seconds), we identified the following strengths. Firstly, resource consumption, which cannot be apportioned during normal operation (e.g. network traffic, storage I/O), can also be included. Secondly, if the resource profiles of services are transparent, they can be used as a criterion for the evaluation of software. So, an incentive for economic resource usage for developers and architects may already be created. According to the literature on Performance Engineering (e.g. [Smit90; BMIS03]) and confirmed by experiences of the BMW Group, most potential for savings are in the design and implementation phases. Finally, resource profiles are valuable inputs for capacity planning and performance analysis. In this context analytical approaches, like Queuing Network Theory, have a number of advantages and are quite well studied. However, due to their complexity and the lack of appropriate input data, they are rarely used in practice. We demonstrated that resource profiles partially fill this gap. In combination with usage forecasts, they facilitate “what-if” analyses and the sizing of infrastructures. In the context of cost allocation “per provided application” and tiered flat rates (see Chapter 2), the predicted capacity requirements are a reasonable criterion for the classification of an application.

Legacy or COTS software components are usually not designed to log service invocations for accounting purposes. We therefore propose linking usage metering to central authentication or authorization services. So, typically the login to an application would be considered as a service invocation. However, this is a trade-off between practicability and flexibility. Furthermore, accounting systems which are designed for an enterprise-wide usage may by default not support hundreds or even of thousands of different services with individual prices and must be adapted accordingly.

For the case of the BMW Group we propose, for reasons of practicability, to retain the cost allocation per application. However, instead of one single flat rate, resource profiles should be used for the determination of rates, based on the expected resource consumption of an application. So, even if the actual consumption cannot be apportioned, the cost driver “infrastructure resource requirements” is considered in cost accounting.

6 Conclusion and outlook

During the project at the BMW Group, we could demonstrate the feasibility of our approach. Nevertheless, the root cause of the difficulties with allocating infrastructure costs is that accounting mechanisms are hardly considered in the architectural guidelines for modern IT systems. Hence, current practice is a per-resource-accounting based on technical metrics, such a CPU seconds. What is still missing, are comprehensive and user-oriented approaches. Predetermined resource profiles are an elegant way out, but do not constitute a general solution. Against the background of advancing virtualisation technologies and the ongoing commercialisation of shared resource pools, we predict a growing demand for uniform standards and protocols to facilitate the allocation of infrastructure resource consumption. Without welt-defined methodologies for usage accounting the foundation for economic efficiency in a competitive world for service provisioning is not given. We strongly believe that in this domain further scientific work could provide valuable contributions for the industrial practice.

References

[Aure97] Aurenz, H.: Controlling verteilter Informationssysteme. Client/Server-Architekturen. Bd. 4, Peter Lang Publishing Group, 1997.

[BMIS03] Balsamo, S.; Marco, A. D.; Inverardi, P.; Simeoni, M.: Software Performance: state of the art and perspectives. http://www.di.univaq.it/adimarco/technicalreports/techReport-perf.pdf, Abruf am 2006-07-13.

[Bert01] Bertleff, C.: Einführung einer IT-Leistungsverrechnung zur Unterstützung des strategischen IT-Controllings. In: H. Heilmann (Hrsg.): Strategisches IT-Controlling. 2001.

[BoGM06] Bolch, G.; Greiner, S.; Meer, H. d.: Queueing Networks and Markov Chains. Wiley-Interscience, 2006.

[DGHJ04] Dibbern, J.; Goles, T.; Hirschheim, R.; Jayatilaka, B.: Information systems outsourcing: a survey and analysis of the literature. In: SIGMIS Database 35 (2004) 4, S. 6-102.

[Harp93] Harper, W. M.: Cost Accounting. 3rd. Aufl., Pitman Publishing, 1993.

[Hein02] Heinrich, L. J.: Informationsmanagement. Bd. 7, Oldenburg, 2002.

[HoMa89] Horváth, P.; Mayer, R.: Prozesskostenrechnung. Der neue Weg zu mehr Kostentransparenz und wirkungsvolleren Unternehmensstrategien. In: Controlling 1 (1989) 4, S. 214-219.

[InteloJ] Intel Corporation: Intel VTune Performance Analyzer. http://www.intel.com/cd/software/products/asmo-na/eng/vtune/index.htm, Abruf am 2006-07-13.

[JaPeoJ] Java Performance Tuning: Tool reports. http://www.javaperformancetuning.com/tools/, Abruf am 2006-07-13.

[KaBr87] Kaplan, R. S.; Bruns, W. J.: Accounting and Management: Field Study Perspectives. Harvard Business School Press, 1987.

[MeAl00] Menascé, D. A.; Almeida, V. A. F.: Scaling for E-Business: Technologies, Models, Performance, and Capacity Planning. Prentice Hall, 2000.

[MeAD04] Menascé, D. A.; Almeida, V. A. F.; Dowdy, L. W.: Performance by Design: Computer Capacity Planning by Example. Prentice Hall, 2004.

[MercoJ] Mercury Interactive Corporation: Mercury LoadRunner. http://www.mercury.com, Abruf am 2006-07-13.

[MoJi98] Mosberger, D.; Jin, T.: httperf - a tool for measuring web server performance. In: SIGMETRICS Performance Evaluation Review 26 (1998) 3, S. 31-37.

[NaAp05] Nagaprabhanjan, B.; Apte, V.: A Tool for Automated Resource Consumption Profiling of Distributed Transactions. In: G. Chakraborty (Hrsg.): Second

International Conference on Distributed Computing and Internet Technology. Bhubaneshwar, India 2005, S. 154-165

[OoOC01] Office of Government Commerce: Service Delivery. Stationery Office Books, 2001.

[OoOC02] Office of Government Commerce: ICT Infrastructure Management. Stationery Office Books, 2002.

[ReLa80] Reiser, M.; Lavenberg, S. S.: Mean-Value Analysis of Closed Multichain Queuing Networks. In: Journal of the ACM 27 (1980) 2, S. 313-322.

[Schw06] Schwichtenberg, H.: Tools and Software Components for the.NET Framework. http://www.dotnetframework.de/dotnet/tools.aspx, Abruf am 2006-07-13.

[SeguoJ] Segue Software Inc.: SilkPerformer. http://www.segue.com/, Abruf am 2006-07-13.

[Smit90] Smith, C. U.: Performance Engineering of Software Systems. Addison-Wesley, 1990.

[SunoJ] Sun Microsystems: Java Pet Store Sample Application. http://java.sun.com/reference/blueprints/, Abruf am 2006-07-13.

[SymaoJ] Symantec Corporation: Application Performance Management. http://www.symantec.com/Products/enterprise?c=prodcat&refId=1021, Abruf am 2006-07-13.

[TheGoJ] The Grinder: A Java Load Testing Framework. http://grinder.sourceforge.net/, Abruf am 2006-07-13.

[WebToJ] WebTrends Inc. http://www.webtrends.com/, Abruf am 2006-07-13.

[Zieh04] Ziehm, O.: Government on Demand. Neue Wege in der Projektfinanzierung. In: I. Corporation (Hrsg.): 7. Deutscher Verwaltungskongress "Effizienter Staat". Berlin 2004