UNDERSTANDING THE PERFORMANCE BEHAVIOR OF A SAP ERP SYSTEM FOR THE USE OF QUEUING MODELS

Stephan Gradl[(a)], Manuel Mayer[(b)], Alexandru Danciu[(c)], Holger Wittges[(d)], Helmut Krcmar[(e)]

[(a) (b) (c) (d) (e)] Chair for Information Systems, Technische Universitaet Muenchen, Boltzmannstrasse 3, 85748 Garching, Germany

[(a) (b) (c) (d) (e)] {Stephan.Gradl, Manuel.Mayer, Danciu, Holger.Wittges, Krcmar}@in.tum.de

ABSTRACT

ERP systems support the management of a company's resources. As a large number of business-relevant processes are supported by ERP systems, the performance and availability of those systems is crucial for the success of a company (Apache Software Foundation 2010). We analyze the response time of 49350 requests. Furthermore, we interpret the system's internal behavior by fetching and analyzing the statistical data. As results we can show that queuing models can be used for evaluating the performance of SAP ERP systems as the response time behavior follows the assumptions of queuing theory, resulting in nearly constant resource consumption per user interaction task, independent of the number of parallel requests. By analyzing the reasons for these results, important insights into the performance behavior of SAP ERP systems for performance analysis and prediction are achieved.

Keywords: ERP, SAP, Performance, Load Test, Measurement, Analysis

1. INTRODUCTION

ERP systems support the management of a company's resources. As a large number of business-relevant processes are supported by ERP systems, the performance and availability of those systems is crucial for the success of a company (Krcmar 2009). In particular, we focus on the performance analysis of the SAP Enterprise Resource Planning (SAP ERP, formerly SAP R/3) application (Schneider 2008). SAP ERP is an integrated backend application with tens of thousands of installations worldwide designed for tracking and managing business processes in midsize and large enterprises. From a technical perspective, this application is built on top of a software integration platform that provides primitives to control the concurrency offered by application server and database server, the layered use of servers, asynchronous messaging, and priority scheduling for certain types of processing. According to Jain (1991), there are several classical approaches for capacity planning and performance evaluation of computer systems like measurement (benchmarking and stress testing), simulation and analytical modeling. To evaluate the performance using simulation techniques, the system has to be modeled. Performance modeling of an ERP system requires deep knowledge about the structure and its performance behavior. To achieve accurate and significant results using queuing models the system has to follow certain performance criteria (Chen et al. 2008):

- The CPU time per user interaction task has to be independent from overall system utilization.
- The CPU utilization has to increase linearly with the number of concurrent load steps.
- The response time has a constant section that is followed by a linearly increasing section, ending in an exponentially increasing behavior.

In the following we describe a case study we performed on an SAP ERP system to analyze the performance behavior of this system when set under heavy parallel. We measure the response time behavior of the system as a black box, and then go a step further and analyze the internal behavior of the ERP system by fetching and interpreting the system's statistical records. Section 2 of this paper provides the research context of this work, while Section 3 provides an overview of the required definitions. In Section 4 we describe how we measure the response time of the system, give a brief overview of the architecture of the system under test and the used benchmark, and point out the method we used to create load. Section 6 then follows with the measurement results, as well as with their interpretation, and the analysis of the statistical records of the ERP system. Finally, in Section 7 we conclude our results and give an overview about the next steps and future work.

2. RELATED WORK

The key literature about performance measurement and analysis of (enterprise) software systems are the books of Jain (1991) and Lilja (2000). These authors describe elaborately the whole process of performance measurement, pointing out what performance is, how it is measured, and which factors affect the performance of a software system. We are basing our work on the definitions made in these books, and adopt them to the fields of ERP. The importance of performance analysis is pointed out by Menascé (2002). An overview of the factors that determine the performance of an application is given by Bailey (2005) and Hollingsworth (2005). For the performance of an ERP system, we refer to (Schneider-Neureither 2004). In this book, the author explains in detail the effect of the SAP architecture and configuration on its performance, focusing on the solution of concrete operational problems. Although the book is written as an administrator manual, it provides a good overview of the factors affecting the SAP system´s performance. An overview of existing SAP benchmarks is given in (Prior 2003).

A scientific approach for the measurement of ERP performance behavior – in this case focusing on the effects of virtualization - presented by Jehle (2009) and Bögelsack (2010). While Jehle is focusing on the response time behavior using a load test, Bögelsack (2008) is analyzing the system´s internal matters, especially the CPU time, for interpreting its effect on the system performance.

Jin (2007) shows a method for performance prediction of legacy information systems. As the internal architecture of the investigated productive information system is not known, the authors used a method that is based on a black box approach for predicting the technical performance of this legacy information system with historical values. This approach combines benchmarking, production system monitoring, and performance modeling (BMM) by analyzing and correlating the performance values derived from the benchmarks and monitoring. Based on the measurements, a model is created and used for the performance prediction.

In (Rolia et al. 2009), an LQN model for the performance prediction of an SAP ERP system is introduced. In this approach the statistical records provided by the SAP system are used for performance analysis and prediction. In addition, the authors also used CPU values gathered from an SAP tool called saposcol. The workload used is based on a sales and distribution scenario, very similar to the workload that is applied in the SAP SD benchmark. Buffers, both from the applications server and the database, having a significant impact on the overall performance, are not taken into account.

3. DEFINITIONS

For measurement the performance of an application, one first has to define what is understood as performance in the given context, and which metric(s) are considered for the representation of an application’s performance. Our understanding of performance is best shown by the following definition, taken from (Schneider-Neureither 2004).

Generally spoken, the performance of a data processing system is its ability to match the requirements in response time and throughput.

As already given by this definition, the most popular metrics for an application’s performance are response (or execution) time and throughput. Nevertheless there are other, like the number of accesses to a special resource of energy need. In our context of ESOA, the response time as defined by Nudd (2000) and shown in figure 1 the time from the moment the request is sent (T1) until the time, when the response is completely received (T3) – is the more relevant metric, as the service calls are considered to be short running by time critical (in contrast to a batch job, where in general the throughput is more relevant than the single response time).

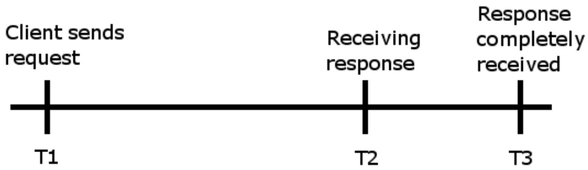

Figure 1: The Structure of the response time (according to Menasce (2002))

Even though it is obvious that the response time and the throughput are connected, in our work we focused on the response time, since the response time is the actual time a user waits while performing a task. On a high dispersion of response times, the throughput could be still eligible, while some high response times are inacceptable (e.g. due to Service Level Agreements).

4. MEASUREMENT

Following (Lilja 2000), the most common benchmark strategy is the fixed-computation approach in which the total time required to execute the benchmark is used as the performance metric. The complementary approach is to fix the amount of time, where the total amount of computation completed in this time period is used as performance metric. The most flexible benchmark strategy is to derive a third dimension from some combination of the execution time and the amount of computation completed within this time. In this way (using this third dimension as performance metric), execution time and computation can be kept variable. The Hierarchical Integration Benchmark (HINT), for instance, uses quality improvements per second as performance metric, defined as a function of the problem being solved by the benchmark program. Table 1 summarizes the strategies that can be used in a benchmark program to exercise the system under test.

For our case study we fixed the amount of computation while measuring the time, needed by the sap ERP System to execute it.

As introduced later on, the benchmark consists of the creation of a material master record in the SAP ERP system. For this task, a WebService in the SAP system has been identified, which has been used for creating load on the system. This service is called in parallel by an own implementation of a Java load generator. For measuring the system behavior, we combine the black box approach described by Kruse (2009), and the glass box approach used by Malik (2010). In the following section, we illustrate the architecture of the system under test, the benchmark, and the load generator.

Table 1: Benchmark Strategies (based on (Lilja 2000))

Time	Computation	Performance Metric
Variable	Fixed	Execution Time
Fixed	Variable	Consumption completed
Variable	Variable	Third dimension

5. SYSTEM ARCHITECTURE

To provide an understanding of the ERP system architecture shown in figure 2, we derive the system components from the ERP process step-by-step by analyzing the recorded trace and the abstraction of the trace entries. These components are described in detail in (Schneider 2008). The process step of calling a program involves many components of the SAP system (see figure 2).

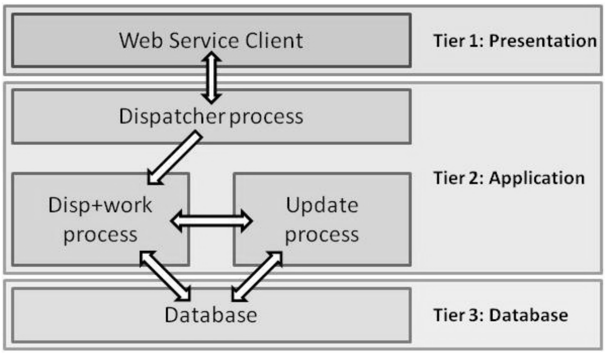

Figure 2: Simplified SAP ERP system architecture

Searching for the program includes access to internal buffers as well as access to the database tables. This access is made by the so called disp+work processes of the SAP system. Such processes are responsible for executing programs, processing user or WebService requests, and accessing the database. Before a request is associated to one of the disp+work processes of the SAP system, a dispatcher process is accessed. The dispatcher process manages all other processes in the SAP system, and his primary task is to assign a user request to a free disp+work process. In our model, we assume the database as a black box.

After the SAP system got the information which program has to be executed, it loads a compiled version of the program from the database and executes it. Sometimes such compiled programs are held in the internal buffers of the SAP system to avoid database accesses.

After the request is processed, the data should be saved to the database. This is done by the disp+work process(es) together with a process called update process. This process receives data and stores it in corresponding database tables.

Simultaneously, a lock on a central table is established, which may be described as a little repository of all available material master records (MMR) within the system. This lock is not set by the disp+work process itself; it triggers a so-called enqueue process. The only task of the enqueue process is to set locks on any tables in the SAP system, and to manage such locks. After the lock was set successfully, the disp+work process can store the data into the central MMR repository.

5.1. Benchmark

For the load run processed in this case study, the Production Planning Integration Case Study (Weidner 2006) has been used. In detail Web Services creating different kinds of material master records, bill of materials, routing, etc. has been chosen. As shown in table 2, these Web Services have an average complexity with read, insert and update statements to the database. Each execution of the case study does not depend on another one and therefore can be executed anytime using the SAP Web Service Interface.

As already introduced, the programs in a SAP ERP System are running on an infrastructure containing dispatching, locking, buffering and database access mechanisms (Schneider 2008). In order to understand the performance of a SAP ERP system, it is necessary to use a workload that uses these components. This leads to different kinds of database queries that characterize the Web Service calls from a technical side.

Table 2: Database accesses for material creation

	DBRows	Read to Buffer	ReqTime
Direct Read	6	222	5134
Sequ. Read	2754	176	6352196
Insert	122	0	12633
Update	1	20	686

Table 2 shows different kinds of database accesses. As the labels Insert and Update are self-explanatory, “Direct reads” are always in the form of “select single” and fetch exactly one row from the database, while queries in the form of “select * from...” are named Sequential reads. The column DBRows shows the number of rows that are fetched directly from the database without being served by the buffers, while the column read2Buffer shows the number of requests that could be served from caches. The column ReqTime contains the time requests took. This entry does not include requests time served by the caches, since these times can be neglected, according to the technical documentation of the Transaction STAD.

In addition, the workload emulates an existing business process, affecting the already mentioned key components (buffers, locks, database accesses, etc) on the technical layer.

For the load test, an ERP installation with an application server and a database server, both hosted on a physical server with 16 GB of Ram and 4 Cores running at 2.8 GHz, has been used. The application server and the database were provided in virtual containers using SUN Solaris Zones. The application server was configured with 30 work processes and the database (MaxDB, version 7.7) with up to 150 parallel connections. For the case study, we used an SAP system with customizing and data of the SAP International Demo and Education System (IDES).

The underlying database for this installation, containing one IDES client, has a size of 220 GB and uses Unicode. The database data files are provided on two internal SAS Discs configured in a performance raid with raid level 0.

5.2. Load Generator

Load generation is done by the own implemented Java application Load Generator. The Load Generator uses threads for parallelization, and the Axis2 framework (Apache Software Foundation 2010) for calling the web service.

A load run is conducted by a stepwise increasing number of parallel service calls. To minimize the amount of overhead, we initialize the payload once, cloning the object tree for each call, and changing the material number. The load procedure calls the service i times in parallel for every step i in the load test (where i = 1 to n, n = the number of maximum parallelism set for the load test), then waits for all results to be stored, and finally increases i by a given step size (step size 1 in our case). Doing this, it is assured that every sequence, which is a load unit of a certain number (i) of parallel requests, is not affected by the request before. For this purpose, a configurable wait time after each sequence has been implemented, too. This short time frame between the sequences is used to fetch the later discussed statistical values from the SAP system, in order to keep these values available and the amount of data, which has to be transferred, as small as possible.

This results in a response time distribution matrix containing the response times for all (i) sequences, and a total number of calls of $m\_calls=(n*(n+1))/2$.

Experiments showed that the initialization of caches on first requests results in non proportional and unpredictable long runtimes. To avoid these “cold start” effects, we perform a settling phase of three times forty requests before starting the measurement. In this way we assure that all caches are initialized, as can be seen by the cache hit statistics provided by the SAP system.

6. RESULTS AND ANALYSIS

In this section we provide two views of the system under test. At first, we interpret the response time behavior, seeing the system as a black box (cf Ludewig (2007)). Afterwards, we switch to the glass box view, taking a deeper look at the system’s internal behavior, analyzing the statistical data, and providing an illustration of where the presented response time behavior originates from.

6.1. Measurement

Figure 3 shows the response time results for five load tests running from one to 140 parallel requests. The diagram contains 49350 response time results, resulting in an easily recognizable behavioral pattern. Using figure 4, we will explain this pattern in the following.

Taking a close look at the response time diagram, one can see that the pattern can be split up in three parts. The first part (marked as 1) is where the number of parallel requests is smaller than the number of available work processes on the ERP system. All requests can be handled by the system in parallel. An increasing number of parallel requests slows down the response times for all requests, as can be seen by the difference between 1.1 (one request) and 1.2 (30 parallel requests). Nevertheless, this slowdown is affecting all requests evenly.

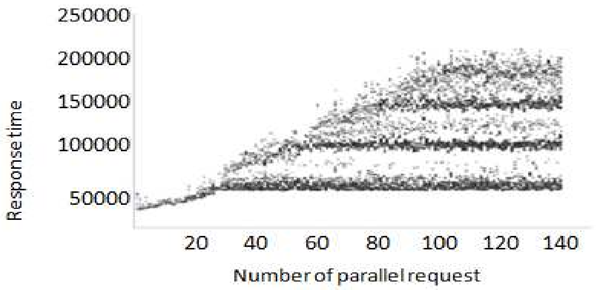

Figure 3: Response time diagram for five load tests

When the number of parallel processes exceeds the number of available work processes, the message queuing used by the ERP system for load balancing becomes visible. The first block of n requests is processed in parallel (where n is the number of available work processes) in a constant time (2), independent of the overall number of requests. This is comprehensible, as the surplus of the requests stays in the queue and thus is not consuming any relevant resources.

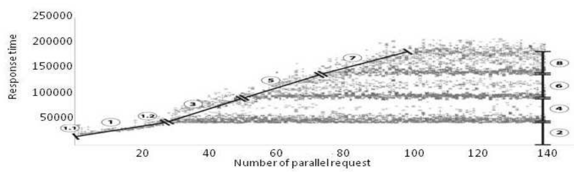

Figure 4: Response time diagram analysis

The surplus of requests is processed after work processes finish the first request. A repeating pattern is recognizable, as the surplus of requests (6) is processed as the requests in section 1. This continues (3 and 7, 4 and 8), up to approximately 100 parallel requests. From this point on, the maximum response time stays constant – due to a timeout of all longer running requests.

As a result we can say that the system’s capacity is at about 100 parallel requests. Passing this limit will result in requests not being successfully answered. But whatever excessive load generated, at least these 100 requests will be successfully processed.

The response time of a single request though is quite unpredictable.

6.2. Glass Box View

To analyze the response time in a glass box view, the relevant components addressed in section 1, and the time slots they use, have to be introduced. The response time in the SAP ERP ABAP stack is quite complex and consists of the following components:

- wait time
- roll in time
- load/generation time
- database time
- enqueue time
- roll out time
- rolled out time
- time in workprocess

As illustrated in figure 1, each request that arrives at the system has to be assigned to a work process in order to get processed. This assignment is done by the dispatcher process. The time for this step is called wait or queue time. The overall time the program is processed by the work process is called time in work process.

As shown in figure 5, this time consists of several components. The first processing step in the work process is to load the process context in its memory. This is called roll in time. If a program calls a remote service, the time, while it has to wait for the response, is also assigned to the roll in time. Then the program has to be loaded into the process memory or generated (compiled) from source code, if it is called the first time. While the request is processed, the work process fetches data from the database. This is aggregated into the metric database time.

The SAP ERP kernel has its own mechanism to control concurrent access to database objects, the so called enqueue process. If a resource is busy, then a work process has to wait until it can be obtained. This slot is called enqueue time. As soon as the work process is finished, the information has to be unloaded from the process memory into the systems shared memory, which is measured by the roll out time. If a request consists of more than 1 work process calls, then the time between the end of the fist call and the beginning of the next call is registered as rolled out time.

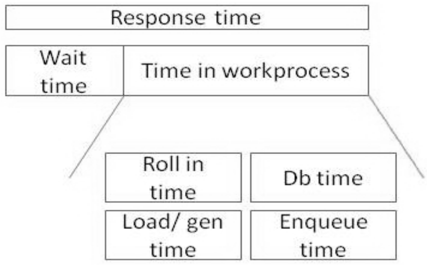

Figure 5: Response time components

These dependencies are shown in figure 3 (response time). In this illustration, the Time in work process consists of the components roll in time, load/generation time, database time, and enqueue time. The roll out time is part of the rolled out time and is not part of the response time, as the response is sent to the client before, in order to reduce the system response time. Different other metrics are not collected, but calculated by the kernel.

The CPU time is a subset of the response time and cannot be assigned to a special component, but is returned by the operating system timer. In the case, of UNIX, the timer works with 100 Hz and consequently the CPU time is always a multiple of 10ms. To analyze the system behavior, the introduced metrics have to be measured. This is done by the SAP ERP kernel completely independent of any ABAP application. The kernel logs a set of performance metrics, like the components of the response time, the program and user name, response size and other important information for performance analysis of the system.

After a request is processed, the work process collects the available information, calculates additional metrics, and stores it in the shared memory. This memory can be accessed by all work processes, and so the performance metrics of the full. As soon as the buffer is full, it is written to binary files on the file system. Every hour a new file named stat is created, while the old one is renamed to stat_<number>. In this way, the statistical records can be accessed as long as the maximum number of stat files is not reached. As soon as this maximum number is exceeded, the oldest file will be deleted. The number of these stat files (and with it the amount of statistical records that are held) is controlled by parameters of the SAP system. For this case study, their values have been increased to hold all the data of a load run. As the logging of statistical records is a standard functionality of the kernel that is always turned on, this method of monitoring can be referred to as non intrusive as mentioned in JAIN, as it does not add an additional load on the system in comparison to a productive usage of the ERP system.

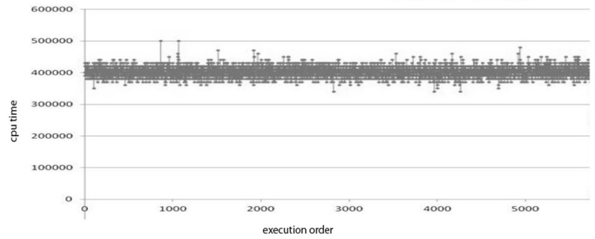

Figure 6: CPU time analysis

The statistical values of each program are displayed using the Transaction STAD in the SAP Gui. To evaluate the big number of these performance values in the presented case study, they have to be exported to a file or a database. For this, several BAPIs exist to deliver the statistical records using RFC function calls. The most complete values are delivered by the BAPI SAPWL_STATREC_DIRECT_READ. As this BAPI is not exported as a web service by the SAP system, it has been copied and modified to meet the requirements for the export as a web service. Thereby, the feature of obtaining the statistical values as soon as the load run is over has been integrated in the presented load generator and analysis tool. To have these performance values easily available, a database has been created to store these values of each load run.

Taking a closer look on the measured values, as discussed in section 5, a clear pattern can be observed. Using the glass box approach, this pattern will be explained by analyzing the process times of each component of the SAP ERP systems kernel. As the system has been warmed up, before the load test has been executed, load/generation time can be neglected, as the ABAP code is already in the corresponding buffer. The same effect can be observed in productive systems, as the size of the buffers is adapted to the expected workload. In addition, the amount of data that has to be rolled in at the beginning of the processing phase is too small to have influence on the systems performance.

The determining factor for the overall response time are the CPU time, total database time, commit time, which is a part of the total db time, and the queuing time. The measured values of these metrics will be introduced in the following to illustrate the performance behavior of the web service used in this case study.

Figure 6 shows the CPU time in nanoseconds according to the timestamp the request has been processed. As the load run has been executed with a growing number of parallel requests, this graph shows that the CPU time, as expected, does not depend on the number of parallel requests. The overall response time is not negatively affected by the CPU time under a growing number of parallel requests.

Figure 7 illustrates the database commit time in nanoseconds; also according to the timestamp the request has been processed. This graph shows a few irregularities, which are created by the log writer mechanism of the underlying database. Besides these irregular values, it does not show any dependencies to the number of parallel requests.

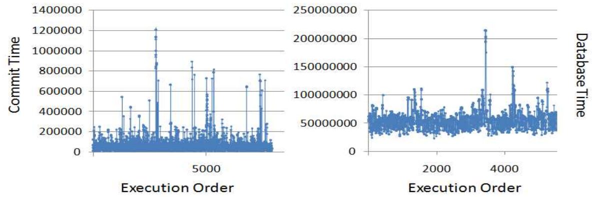

Figure 7: Commit time and database time

The total database time is shown in the right part of figure 7. There is no indication that the performance of the database component does significantly change with the number of parallel requests. As the ERP system has a certain number of work processes, it has been assumed that the database response time does not change significantly with a rising number of parallel requests to the SAP ERP system, as there are not more requests arriving at the database than work processes in the system exist. But even in the starting phase, where less parallel requests had to be served by the system and the database, the performance of the database is in a certain frame.

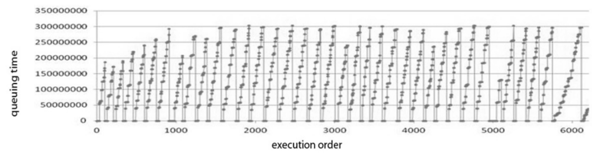

Figure 8: Queuing time

An explanation of the measured values shown in figure 4 is found in figure 8 showing the queuing time in nanoseconds; according to the timestamp the request has been processed.

The graph shows a significant dependency to the number of parallel requests. As in the stating phase the queuing time is small, it grows with more requests sent to the system. With 30 work processes configured, the first 30 requests had no or a negligible queuing time, while the remaining requests had to wait until one of the first requests has been finished. Consequently, the queuing time rises with the number of parallel requests until the sequence is finished.

7. LIMITATIONS

This case study does not regard the usage of different technical users on the SAP system. However, in a well configured productive SAP system, the shared memory that is intended to hold the user contexts is big enough to hold all data in shared memory without the need to put some data into a slower memory area. Therefore, the simplification of reducing the amount of different users does not reduce the significance and validity of the case study's results.

In addition, the impact of a suboptimal system configuration to system response times is not part of this work. Using a non optimal resource distribution, several system components may respond differently compared to the response times and components' behavior in general measured in this case study. For instance, in some special cases the work process is assigned to a user as long as he is logged into the system. This work aims at understanding the general concept of the response time of an ERP system that is well parameterized, and not at the analysis of performance problems.

Equally, the different behaviors of ERP systems that are used for development purposes, so called "DEV" Systems, are not regarded in this case study.

A challenging task when analyzing complex software systems is the decision how many components should be integrated in the analysis. Even in this defined example, the paper demonstrates that a lot of data is gathered from several components in the SAP system and that even the database can be described in a more detailed way. In (Gradl et al. 2010) an eight level architecture was presented to limit the effort of building the architecture of the SAP system. By analyzing the SAP system and its traces, it was discovered that the lowest level is the response time level of the database. As the SAP system does not provide more detailed information about the database, the database has been treated as black box. Response time values of the database are derived directly from the SAP system.

8. CONCLUSION

On the first view, the response time behavior of the analyzed service seems to be complex. But on a second view it reveals a quite predictable behavior. Using a queuing mechanism, the SAP system processes several requests in parallel, and at the same time it regulates the resource consumption by limiting the number of parallel processed requests to the number of work processes available. This leads to a stable environment not being affected by the amount of parallel requests, and a guaranteed response time for some of the requests.

Taking a look inside the system, it can be seen that the resource consumption behind the messaging layer is quite constant. Considering the queuing, this is not surprising. Only the commit time shows some volatility, caused by the fluctuation of parallel processes accessing the database. This volatility of the commit time also causes the jitter in the response time diagram.

In summary we draw the conclusion, that the dispatching time represents the difference between response times, while the processing time is constant independent of how many parallel requests are processed. While this sounds obvious at first, it is an important validation for performance analysis and prediction.

As next steps, we use the information acquired in this work and performance data gathered to parameterize a layered queuing model based on (Gradl et al. 2010) to predict the performance of a SAP ERP system via simulation (Woodside 2002).

REFERENCES

Apache Software Foundation "Apache Axis2/Java," 2010.

Bailey, D.H., and Snavely, A. "Performance Modeling: Understanding the Present and Predicting the Future," in: *Euro-Par 2005 Parallel Processing*, Springer, Berlin / Heidelberg, 2005, pp. 185-195.

Bögelsack, A., Jehle, H., Wittges, H., Schmidl, J., and Krcmar, H. "An Approach to Simulate Enterprise Resource Planning Systems," in: *6th International Workshop on Modelling, Simulation, Verification and Validation of Enterprise Information Systems, MSVVEIS-2008, In conjunction with ICEIS 2008*, U. Ultes-Nitsche, D. Moldt and J.C. Augusto (eds.), INSTICC PRESS, Barcelona, Spain, 2008, pp. 160-169.

Bögelsack, A., Krcmar, H., and Wittges, H. "Performance Overhead of Paravirtualization on an Exemplary ERP System," in: *12th International Conference on Enterprise Information Systems*, Funchal, Madeira, Portugal, 2010.

Chen, S.G., and Lin, Y.K. "Performance analysis for Enterprise Resource Planning systems," Industrial Engineering and Engineering Management, 2008. IEEM 2008. IEEE International Conference on, 2008, pp. 63-67.

Gradl, S., Mayer, M., Wittges, H., and Krcmar, H. "Modeling ERP Business Processes Using Layered Queueing Networks," in: *12th International Conference on Enterprise Information Systems*, Funchal, Portugal, 2010.

Hollingsworth, J.K., Snavely, A., Sbaraglia, S., and Ekanadham, K. "EMPS: An Environment for Memory Performance Studies," in: *Proceedings of the 19th IEEE International Parallel and Distributed Processing Symposium (IPDPS'05) - Workshop 10 - Volume 11*, IEEE Computer Society, 2005, p. 223.222.

Jain, R. *The art of computer systems performance analysis* Wiley, New York, 1991.

Jehle, H. "Performance-Messung eines Portalsystems in virtualisierter Umgebung am Fallbeispiel SAP," in: *CVLBA Workshop 2009. 3. Workshop des Centers for Very Large Business Applications (CVLBA)*, Arndt, H.-K.; Krcmar, H., Magdeburg, Deutschland, 2009.

Jin, Y., Tang, A., Han, J., and Liu, Y. "Performance Evaluation and Prediction for Legacy Information Systems," in: *ICSE´07*, IEEE, Minneapolis, 2007.

Krcmar, H. *Informationsmanagement* Springer-Verlag New York, Inc., 2009.

Kruse, H.G. *Leistungsbewertung bei Computersystemen*, (1. ed.) Springer-Verlag, Berlin, Heidelberg, 2009.

Lilja, D.J. *Measuring Computer Performance - A practitioner's guide* Cambridge University Press, 2000.

Ludewig, J., and Lichter, H. *Software Engineering - Grundlagen, Menschen, Prozesse, Techniken*, (1. ed.) dpunkt.verlag GmbH, Heidelberg, 2007, p. 618.

Malik, H. "A methodology to support load test analysis," in: *Proceedings of the 32nd ACM/IEEE International Conference on Software Engineering - Volume 2*, ACM, Cape Town, South Africa, 2010, pp. 421-424.

Menascé, D.A. "Software, performance, or engineering?," in: *Proceedings of the 3rd international workshop on Software and performance*, ACM, Rome, Italy, 2002, pp. 239-242.

Menasce, D.A., and Almeida, V.A.F. *Capacity Planning for Web Services: metrics, models, and methods* Prentice Hall, Upper Saddle River, NJ, Upper Saddle River, NJ, 2002.

Nudd, G.R., Kerbyson, D.J., Papaefstathiou, E., Perry, S.C., Harper, J.S., and Wilcox, D.V. "Pace - A Toolset for the Performance Prediction of Parallel and Distributed Systems," *International Journal of High Performance Computing Applications* (14:3), August 1, 2000 2000, pp 228-251.

Prior, D. "Who Sets the Pace in the SAP Performance 'Olympics'?," *Gartner*), 24.2.2003 2003, p 6.

Rolia, J., Casale, G., Krishnamurthy, D., Dawson, S., and Kraft, S. "Predictive modelling of SAP ERP applications: challenges and solutions," in: *Proceedings of the Fourth International ICST Conference on Performance Evaluation Methodologies and Tools*, ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), Pisa, Italy, 2009, pp. 1-9.

Schneider-Neureither, A. *Optimierung von SAP-Systemlandschaften*, (1. ed.) Galileo Press, Bonn, 2004.

Schneider, T. *SAP-Performanceoptimierung*, (5 ed.) Galileo Press, Bonn, 2008, pp. 247-249.

Weidner, S. "Integrations-Fallstudie PP (SAP ECC 5.0)," 2006.

Woodside, M. "Tutorial Introduction to Layered Modeling of Software Performance," 2002.

OBJECT-ORIENTED MODEL OF FIXED-BED DRYING OF COFFEE BERRIES

Emílio de Souza Santos[(a)], Paulo Cesar Corrêa[(b)], Brian Lynn Steward[(c)], Daniel Marçal de Queiroz[(d)]

[(a)] Regional Development Analyst, Agricultural and Environmental Engineer, CODEVASF, Brasilia/DF – Brazil.
[(b), (d)] Associate Professor, Department of Agricultural Engineering, Viçosa Federal University, Viçosa/MG – Brazil.
[(c)] Associate Professor, Dep. of Agricultural and Biosystem Engineering, Iowa State University, Ames/IA – USA.

[(a)]emilio_ss@hotmail.com, [(b)]copace@ufv.br, [(c)]bsteward@iastate.edu, [(d)]queiroz@ufv.br

ABSTRACT
The aims of this work were to validate a new model of a fixed-bed drying process with experimental data, compare this model with a traditional one (MSU Model), and analyze the sensitivity of each model parameter. A low level model of a thin-layer was developed, based on mass and heat balance between the drying air and the product. It also considered the product physical properties variation during drying. Thus a high level model was created by connecting four thin-layer models, in order to represent a thick layer. The proposed model was validated by the analysis of model performance index between the experimental data and simulated results. Finally, the efficiency indexes of the proposed model and the MSU model were compared. The models and the simulation were done using OpenModelica® 1.6.0, based on the Modelica language. The proposed model had shown good performance indexes compared with the MSU model.

Keywords: Modelica, physical properties, MSU Model, model performance.

1. INTRODUCTION

After harvest, coffee is very perishable due to the high moisture and sugar content. Thus, coffee passes through a drying operation, to enable safe storage and minimize quality degradation prior to subsequent processing. Drying is defined as an operation of moisture removal through to simultaneous heat and mass transfer (Henderson et al. 1997). It is also one of the preservation methods of agricultural products which are most often used and is the most energy-intensive process in industry (Dincer 1998). Moreover, drying is a difficult food processing operation mainly because of undesirable changes in the quality of the dried product. Such changes can occur because of substantial changes in the product's chemical and physical properties during the drying process.

In a drying system, two fundamental tasks must be performed. First, energy must be provided to the product to be dried so that moisture contained in the product is vaporized. Second, the vaporized moisture must be removed by the drying system (Henderson et al. 1997). Thus, air used to dry the product, after passing through a thin-layer of product, experiences a decrease in temperature and an increase in its humidity ratio. Consequently, thick-layer drying is a complex dynamic system, considering that the drying kinetics will change due to the changes in air and product properties through time and space.

In order to reduce costs and robustly control drying systems, simulation of the process is important, as a mean to understand the system dynamics and then optimize the system. Many models have been developed to simulate fixed-bed drying. A commonly used model is the MSU model (Michigan State University). This model consists of a system of four differential equations resulting from mass and energy balances of a control volume (Dalpasquale et al. 2008). The MSU model is easy to apply to different agricultural products because it assumes that some product properties is equal to water properties, e.g, specific heat. However, this model tends to overestimate the drying rate (Brooker et al. 1992), due to the model assumptions.

Currently, with the rapid improvement of computer and software technology and the widespread availability of computational resources, high fidelity models can be implemented with simulation that more closely represent reality. Some recent examples of new drying models include the work of Izadifar and Mowla (2003), who considered the theory of simultaneous mass and heat convection and internal mass diffusion in their model; Guiné et al. (2007) who based their model on the liquid diffusion theory, considering the product shrinkage and physical properties variation during drying; and Lecorvaiser et al. (2010), who considered the dynamic phenomena of air turbulence.

In addition, the dynamics of the drying system are coupled with those of various components that are necessary for the operation of the entire system including motors, fans, conveyors, heaters and control devices. Each component performs a specific operation that effect the entire system and the system affects the operation of each component. The simulation of this type of system is usually performed by simulating each component individually. This approach is taken due to the difficulty of developing an overall system model that represents many physical domains including the electrical, mechanical, thermodynamic, hydraulic, chemical, and control domains.

Recently though, modeling technologies have been advanced to ease the development of complex, multi-domain, physical models. For example, *Modelica* is an object-oriented equation-based programming language which enables multi-domain modeling, meaning that model components corresponding to physical objects from several different domains that can be described and connected (Fritzson, 2003). Therefore, the Modelica language is promising for drying system modeling, since this language can enable modeling of all the physical domains and mechanisms in this type of system.

The aims of this work were to: (1) model the drying process of a fixed-bed coffee dryer considering the variation of the product physical properties and drying air flow; (2) validate the model by comparing the simulate results with experimental data; (3) compare this model with the MSU Model; and (4) analyze parameter sensitivity in order to understand and simplify the proposed model.

2. METHODOLOGY

The present work was conducted in the Laboratory of Evaluation of Physical Properties and Quality of Agricultural Products in the Brazilian Grain Storage Training Center – CENTREINAR, Federal University of Viçosa, Viçosa, MG, Brazil.

2.1. Modeling

The proposed model to describe the fixed-bed coffee dryer was built *Modelica* language. Real physical objects are represented on *Modelica* language as *object* block. Each object is an instance of a specific *class*, also called *model*. Furthermore, *models* contain both the state of the object, represented by *variables*, and the behavior of the object, represented by *equations*.

Object-oriented language enables a process known as *inheritance*. *Base-class* states (variables) and behaviors (equations) can be extended to a *sub-class*. Thus, the model hierarchy is organized by levels, where models are classified as a higher level than the inherited model. Figure 1 shows the schema of the proposed fixed-bed-dryer model and the model hierarchy.

The coffee fixed-bed was considered to be a thick layer composed of a finite number of thin layers. Each thin layer is a control volume that has a fixed transversal area A and a variable thickness L. The drying kinetics in each control volume were described using thin layer drying theory and were modeled as a Modelica model class.

Modelica model classes are connected with connector class which represents the conservation relations between model classes. In this case, the connector variables were air properties. The connector`s flow variable is the velocity ($V$) which is proportional to mass flow rate. The connector`s potential (non-flow) variables are the temperature (T), pressure (p) and relative humidity (rh) of the air. The sub-index a denotes an input variable and the sub-index b denotes an output variable. Figure 2 shows a scheme of the thin layer model (*DryLayer*) with the inlet and outlet connector.

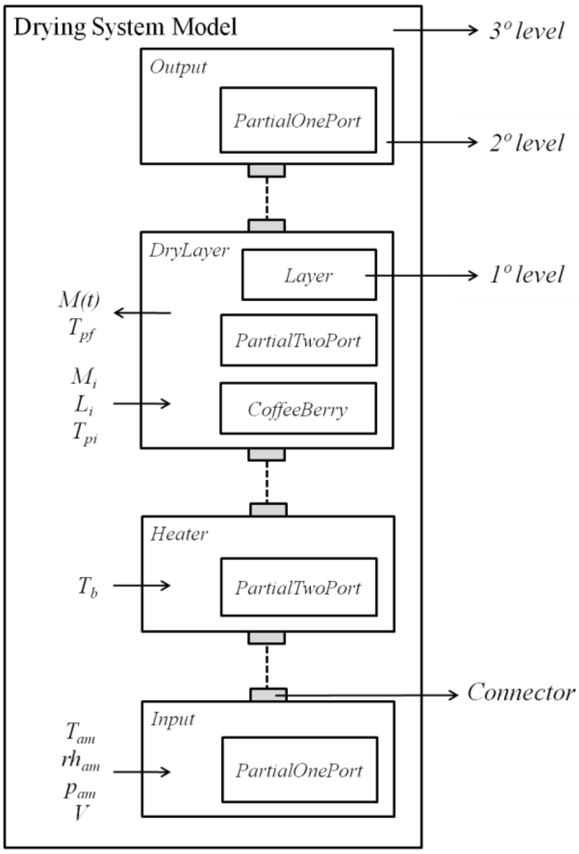

Figure 1: Hierarchy schema of the thick layer drying

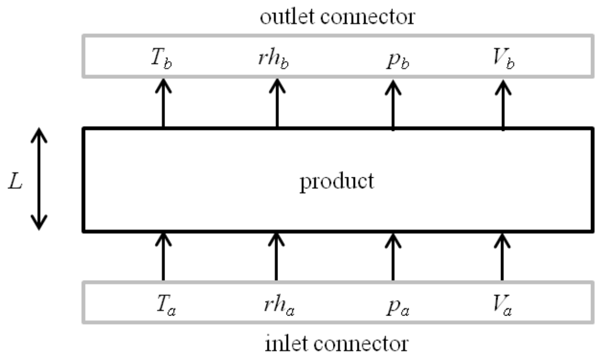

Figure 2: Control volume and air connectors of the *DryLayer* model.

The first-level-models were created containing the air psychometrics equations and properties. The second-level-model to represent the air inlet and outlet are named *Input* and *Output*, respectively. These models need only a single connector, inlet or outlet. Thus, a first-level-model, named *PartialOnePort*, was created containing the psychometrics equations and a connector. The others second-level-models need a pair of connectors (inlet-outlet). Thus, a first level model was created, named *PartialTwoPort*.

The saturation vapor pressure was described as function of temperature by an empirical equation proposed by Keenan and Keyes (Henderson et al. 1997).

$$\ln\left(\frac{p_{vsat}}{R'}\right) = \frac{A + BT + CT^2 + DT^3 + ET^4}{FT - GT^2} \quad (1)$$

where: p_{vsat} is the air saturation vapor pressure, Pa; T is the air temperature, K; R', A, B, C, D , F and G are the adjusted empirical coefficients.

The air vapor pressure, humidity ratio, enthalpy and density were calculated using basic psychometrics equations. These equations were formulated based on Dalton`s law, mass and energy conservation principles.

$$H = \frac{p_v}{1.605\,(p - p_v)} \quad (2)$$

$$\rho_{ar} = \frac{(1+H)\,(p - p_v)\,MM_{H_2O}}{R\,T} \quad (3)$$

$$p_v = rh\; p_{vsat} \quad (4)$$

$$h = c_{ar}(T - T_0) + H\,[h_{g,0} + c_v(T - T_0)] \quad (5)$$

where: H is the air humidity ratio, kg of vapor per kg of dry air; p is the air absolute pressure, Pa; p_v is the air vapor pressure, Pa; ρ_{ar} is the humid air density, kg m^{-3}; MM_{H_2O} is the water molar mass, 18.02 kg kmol^{-1}; R is the universal gas constant, 8314 J mol^{-1} K^{-1}; rh is the air relative humidity, dimensionless; h is the humid air enthalpy, kJ kg^{-1}; $h_{g,0}$ is the water latent heat at the reference temperature, 2502.5352 kJ kg^{-1}; T_0 is the reference temperature, 273.15 K; c_{ar} is the dry air specific heat, 1.0069 kJ kg^{-1} K^{-1}; and c_v is the vapor specific heat, 1.8757 kJ kg^{-1} K^{-1}.

The air and vapor mass flows are calculated in the first level models. These variables are used in the mass and energy balance equations.

$$m'' = \rho_{ar}\,V \quad (6)$$

$$m''_v = \left(\frac{H}{H+1}\right) m'' \quad (7)$$

where: m'' is the air mass flow, kg m^{-2} s^{-1}; m_v'' is the vapor mass flow, kg m^{-2} s^{-1}; e V is the air velocity, m s^{-1}.

The first-level-model named *CoffeeBerry* was created containing the mathematical equation that describes the product physical properties as functions of its moisture content. This model is stored in a Package named *Product*, which can be used to storage many types of product models.

The coffee volume, density, porosity and specific heat were described using the equations 8 to 11, which were developed by Junior (2001). Assuming the coffee berry to be a perfect sphere, the product average diameter can be calculated using equation 12.

$$v = 10^{-9}\,(621.46 + 152{,}78\,M + 12.417\,M^2) \quad (8)$$

$$\rho_p = 420.8490 + 198.8201\,M - 53.8475\,M^2 \quad (9)$$

$$\varepsilon = 10^{-3}(432.324 + 114.307\,M - 32.317\,M^2) \quad (10)$$

$$c_p = 0.9447 + 3.6197\,M - 1.9920\,M^2 \quad (11)$$

$$v = \frac{\pi\, d_p^{\;3}}{6} \quad (12)$$

where: M is the product moisture content, kg of water per kg dry mass (d.b.); v is the product volume, m^3; ρ_p is the product density, kg m^{-3}; ε is the product porosity, dimensionless; d_p is the product average diameter, m; and c_p is the product specific heat, kJ kg^{-1} K^{-1}.

The drying constant was calculated using the equation 13, proposed by Young and Dickens (Henderson et al. 1997). This equation relates a general drying constant to a referential one, obtained experimentally. The empirical equation of Junior (2001) was used as the referential drying constant. The saturation vapor pressure ratio in equation 13 was assumed to be equal to 1 due to the equation 14 already considering the variation of relative humidity. The referential velocity is equal to 0.2166 m s^{-1}, the value used in Junior's experiment.

$$k = {}^{1}\!/_{3600}\; k_r \left({}^{p_{sat}}\!/_{p_{satr}}\right)^{0.46} \left({}^{V}\!/_{V_r}\right)^{0.7} \quad (13)$$

$$\begin{aligned} k_r = & -0.1196 + 1.4180\,10^{-3}\,T_c + 6.9938\,10^{-5}\,T_c^2 \\ & + 0.6545\,rh - 0.5369\,rh^2 - 7.5170\,10^{-3}\,T_c\,rh \end{aligned} \quad (14)$$

where: k is the drying constant, s^{-1}; k_r is the referential drying constant, h^{-1}; p_{satr} is the referential saturation vapor pressure, Pa; V_r is the referential air velocity, m s^{-1}; T_c air temperature in Celsius degree , °C.

The product water latent heat was described using equation 15, adjusted by Junior (2001). The Modified Henderson equation (16), adjusted by Correa et al. (2010), was used to describe the hygroscopic equilibrium between the product and the air.

$$\begin{aligned} h_{fg} = & (2502.49 - 2.43\,T_c) \\ & (1 + 7.7866\,10^{6}\,\exp(-19.6621\,M^{0.0499})) \end{aligned} \quad (15)$$

$$1 - rh_e = \exp(-0.0001\,(46.8549 + T_c)\; M_e^{1.8299}) \qquad (16)$$

where: h_{fg} is the water latent heat inside the product, kJ kg^{-1}; M_e is the product equilibrium moisture content, d.b.; and rh_e is the equilibrium relative humidity, dimensionless.

A first-level-model named *Layer* was created to define initial values of the variables: the initial layer thickness (L_i), initial moisture content (M_i) and initial product temperature (T_{pi}). The initial layer thickness of the control volume has to be as small as possible, in order for the thin layer drying theory to be applicable.

The *Input* model extended the *PartialOnePort* model and has the structure to receive the boundary variables of the air. The *Output* model only needs to extend the *PartialOnePort* model.

The *Heater* model was created to represent the air heater. This model extended the *PartialTwoPort* model and has the structure to receive the variable T_b, which represent the temperature that the air is heated (drying temperature). As result, all outlet air variables are calculated by the *Heater* model.

The *DryLayer* model was created to represent the control volume of product. This model extended the *PartialTwoPort*, *CoffeeBerry* and *Layer* models and has the mass and energy balance of the drying process.

The exponential equation, proposed by Sherwood (Henderson et al. 1997), was used to describe the thin layer drying of the product. This equation assumes that the drying rate is proportional to the difference between the moisture content and the equilibrium moisture content, proportionality given by the product drying constant.

$$\frac{dM}{dt} = -k(M - M_e) \qquad (17)$$

where: dM/dt is the drying rate, s^{-1}.

The heat balance equation considering on *DryLayer* model neglects the heat transfer due to conduction between the particles and radiation between the particles and the dryer's walls. Moreover, it was assumed that all heat transferred by the air is used to the product water evaporation and to heat the product. Thus the changes in air properties can be related to the variation on product moisture content, using the mass (18) and heat (19) balance equations. The convective heat transfer coefficient was calculated using the Barker`s empiric equation (21), presented by Brooker et al. (1992).

$$m''_{vb} - m''_{va} = -\frac{dM}{dt}\,\frac{\rho_p\, L}{1+M} \qquad (18)$$

$$m''_b\, h_b - m''_a\, h_a = \frac{dM}{dt}\,\frac{\rho_p\, L}{1+M}\, h_{fg} - \rho_p\, L\, c_p\, \frac{dT_p}{dt} \qquad (19)$$

$$m''_b\, h_b - m''_a\, h_a = -\,h'\,(T_a - T_p) \qquad (20)$$

$$h' = 0{,}9918\, c_{ar}\, m''_a \left(\frac{3600\, d_p\, m''_a}{0{,}06175 + 16{,}5\,10^{-5}\, T_c} \right)^{-0{,}34} \qquad (21)$$

where: dT_p/dt is the product temperature rate, K s^{-1}; T_p is the product temperature, K; and h' is the heat transfer coefficient W m^{-2} K^{-1}.

The pressure variation through the product layer was calculated using the equation 22, which assumes that the total pressure variation is equal to the vapor pressure variation plus the friction loss. The friction loss was calculated using the Darcy equation (23) and the friction factor was calculated using the Ergun equation (24) (Henderson et al. 1997).

$$p_b - p_a = (p_{vb} - p_{va}) - p_f \qquad (22)$$

$$p_f = f\, \rho_{ar,a}\, \frac{L}{d_p}\, \frac{V_a^2}{2} \qquad (23)$$

$$f = \left(\frac{1-\varepsilon}{\varepsilon^3} \right) \left(3.5 + \frac{300\,(1-\varepsilon)}{Re} \right) \qquad (24)$$

$$Re = \frac{\rho_{ar,a}\, V_a\, d_p}{\mu_{ar,a}} \qquad (25)$$

where: p_f is the friction loss, Pa; f is the friction factor, dimensionless; Re is the modified Reynolds number, dimensionless; μ_{ar} is the air dynamic viscosity, Pa s.

The layer thickness variation was calculated using the equation 26. This equation considers the dry mass conservation and that the volume shrinkage only happens on the thickness direction, due to the fixed transversal area of the dryer. The sub-index *i* denotes initial.

$$\frac{\rho_v\, L}{1+M} = \frac{\rho_{v,i}\, L_i}{1+M_i} \qquad (26)$$

Many *DryLayer* model can be connect to form a third-level-model that represents a thick fixed layer. The model was implemented in *Modelica* language and simulated using the *OpenModelica*® 1.6.0 package.

2.2. Experimental Data

The experiment was conducted in a prototype dryer consisted by three chambers with equal dimension of 0.57 x 0.35 x 0.64 m. It was used coffee berries (*Coffea arábica L.*) variety *Mundo Novo*, manually harvested and pre-dried with ambient air.

The coffee was dried with dryer air (initial inlet air) with temperature at three levels: 40, 50, 60 ^oC. Three replicates were done for each temperature, resulting in nine experimental tests. Each test had a specific condition of ambient air and different initial moisture content of the product, presented on Table 1.

The coffee samples were withdrawn at predetermined periods at the heights of 0.10, 0.25, 0.40 and 0.55 m, in order to determine the product moisture content in different layers. For the simulation, it was assumed that each sample was withdrawn at the center of the layer, so the first and third layers have 0.20 m of thickness and the second and forth layers have 0.10 m of thickness.

Table 1: Drying and ambient conditions for each experimental tests.

Test	T_s **°C**	T_{am} **°C**	rh_{am} **%**	p_{am} **kPa**	V **m/min**	M_i **d.b.%**
40.1	40	23.5	55.9	93.843	8.60	17.91
40.2	40	25.4	52.7	93.683	8.81	20.41
40.3	40	20.2	67.7	94.216	8.27	20.25
50.1	50	22.9	48.0	94.376	9.20	38.56
50.2	50	23.2	62.2	93.750	8.82	18.56
50.3	50	19.8	69.6	94.136	8.72	18.10
60.1	60	22.9	57.7	93.790	7.10	17.45
60.3	60	16.7	45.7	94.083	8.27	32.32
60.3	60	24.3	43.9	93.817	8.81	19.70

where: T_s is the drying air temperature; T_{am} is the ambient temperature; rh_{am} is the ambient air relative humidity; p_{am} is the ambient air absolute pressure; V is the inlet air velocity; M_i is the product average initial moisture content.

2.3. Model Validation

The model statistical performance was evaluated by analysis of the relative standard deviation (*RSD*) and the performance index (27). This index evaluates the model performance by multiplying one dimensionless number that corresponds to the model precision, given by correlation coefficient (*r*), and another that corresponds to the model accuracy, given by agreement index (28) (Willmoot et al. 1985).

$$i = r\,d \tag{27}$$

$$d = 1 - \left[\frac{\sum_{j=1}^{n} \left(Y_j - \hat{Y}_j \right)^2}{\sum_{j=1}^{n} \left(\left| Y_j - \bar{Y} \right| + \left| \hat{Y}_j - \bar{Y} \right| \right)^2} \right] \tag{28}$$

where: i is the performance index; r is the correlation coefficient; d is the agreement index; Y is the observed data; $\hat{Y}$ is the model-predicted data; $\ddot{Y}$ is the average value of observed data; e n is the number of observed data.

2.4. Model comparison

The proposed model performance was compared with the MSU model, in order to investigate the validity of the new model. The performance comparison between the models was done by analyzing the performance index of each model.

The MSU model is completely described by Brooker et al. (1992), and was implemented in *Modelica* language and simulated with the same conditions of the proposed model using the *OpenModelica*® 1.6.0 package.

If compared with the proposed model, the MSU model have the following particularities: constant physical properties; assumption of water latent heat being equal to the free water; neglecting of pressure variation due to the increasing of vapor pressure through the drying; and neglecting of the layer volume shrinkage.

2.5. Sensitivity Analysis

Model sensitivity to parameter variation was evaluated using the differential sensitivity analysis method. In this method, all model parameters were classified by means of the sensitivity coefficient (29). The sensitivity coefficient represents the ratio between the output variable variation (response) and the analyzed parameter variation (perturbation), while all other parameters remain constant (Hamby 1994). The model result while all parameters are held constant is defined as the "base case". The sensitivity coefficient was calculated to response of M, T_p, rh and T of all layers, and a global value S_t was calculated as an average value of all calculated sensitivity coefficients. The analysis was done considering a variation of ± 20% of each model`s parameter.

$$S_{X,\beta} = \frac{\left| X - X_i \right|}{\left| \beta - \beta_i \right|} \frac{\beta}{X} \tag{29}$$

where: $S_{X,\beta,}$ is the sensitivity coefficient response of X to the β variation; X is the output variable value to the base case; X_i is the output variable value to the β variation; β is the value of the analyzed parameter to the base case; β_i is the varied value of the analyzed parameter.

3. RESULTS

3.1. Simulation Results

Figure 3 shows the experimental observed values and predicted simulate results of moisture content for test 40.3, 50.3 and 60.3.

It can be observed in Figure 3 that the drying capacity increases when the temperature increases. This behavior occurs due to the increase of the product internal water diffusion and the increase of vapor pressure gradient between the water layer surface and the drying air.

Vossen (1979) indicated that a moisture content lower than 14 % (d.b.) is essential to ensuring safe storage of coffee. Furthermore, a low moisture content of coffee berries is important in the husking process. For the experimental conditions of the 40.3° C test, the 12 hour drying time was not enough to ensure a moisture content of 14 % (d.b.) in all layers. It is recommended for the industrial coffee drying process that the process end when the last product layer reaches the 14 % (d.b.) moisture content.

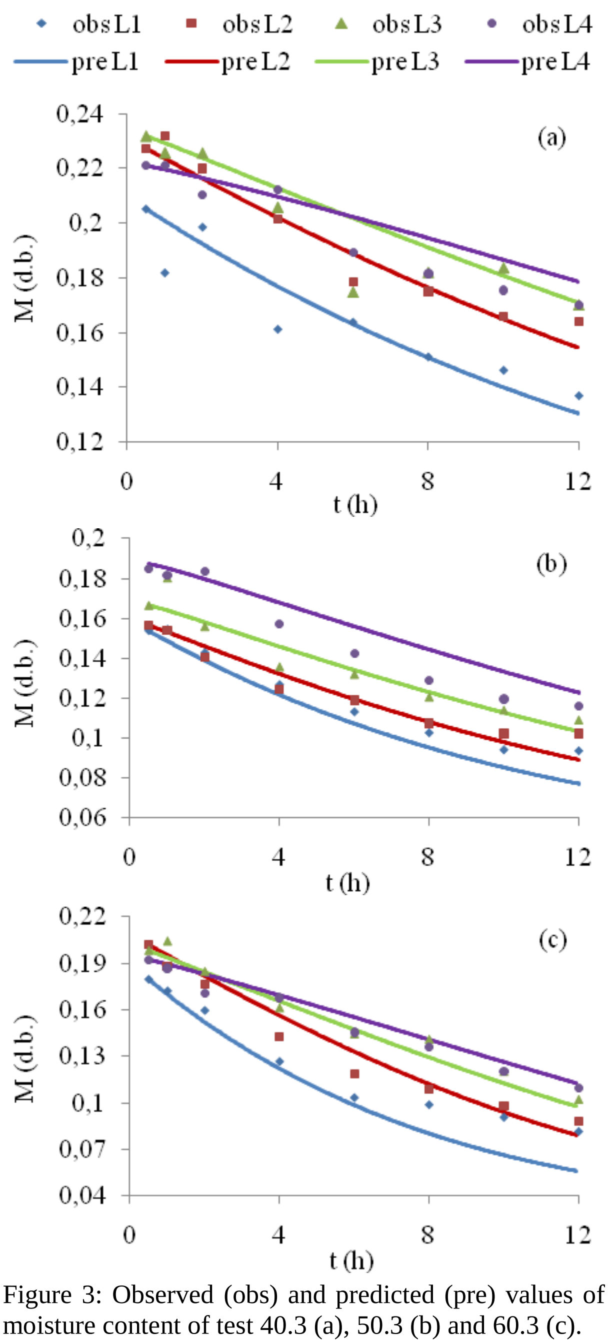

Figure 3: Observed (obs) and predicted (pre) values of moisture content of test 40.3 (a), 50.3 (b) and 60.3 (c).

Another important parameter to evaluate for process quality is the product temperature (T_p). Clarke and Macrae (1987) stated that if the coffee berry reaches a temperature of 40°C, the product suffers physical and chemical changes that will decrease the quality of the coffee beverage. Thus, it is recommended that the maximum drying temperature for static dryers be 40°C (Sfredo et al. 2005). Figure 4 shows the simulate results of product temperature for the test 40.3, 50.3 and 60.3.

It can be observed in Figure 4 that T_p reached a value higher than 40 °C for layer 1, 2 and 4 in the 60.3° C test. Based only on the simulation results, the use of a temperature higher than 60 °C for drying will generate a low quality product. In the 50.3° C test, only T_p of layer 2 reaches a value higher than 40 °C. Probably the use a temperature of 50 °C can be used to drying coffee depending on the drying conditions and system control.

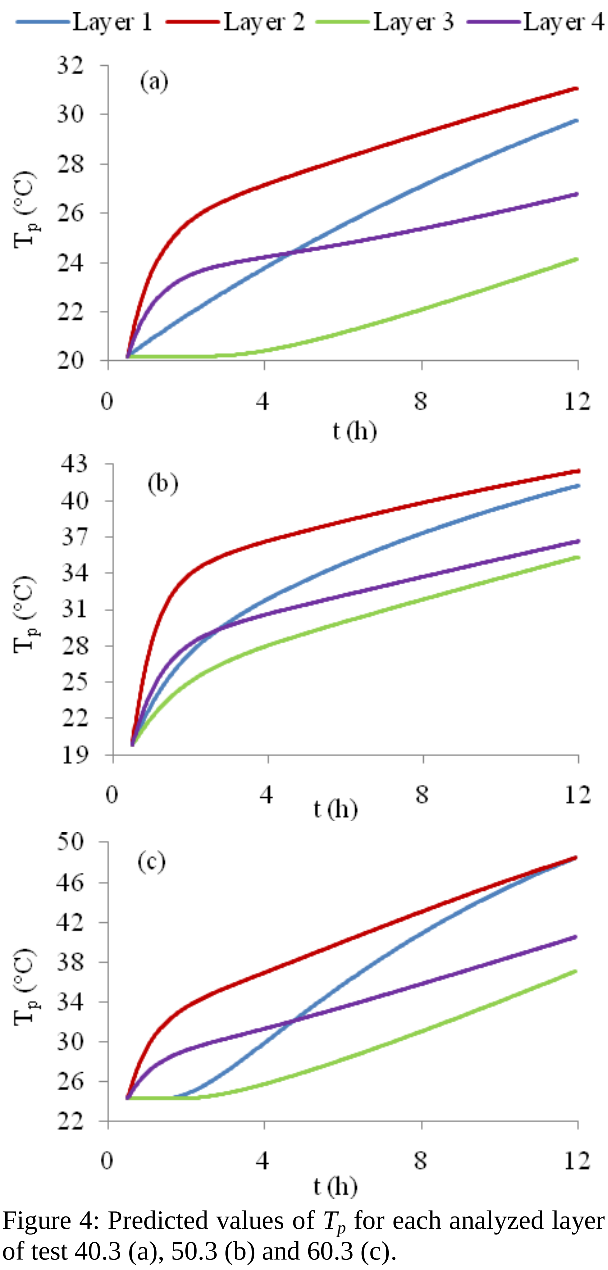

Figure 4: Predicted values of T_p for each analyzed layer of test 40.3 (a), 50.3 (b) and 60.3 (c).

The temperature of small particles is hard to measure incisively due to the size of the measurement equipment. Usually the product surface temperature or the transfer fluid temperature is used as a quality process parameter. In this contest the simulation results of product temperature can be used as a process control.

3.2. Validation

Table 2 shows the agreement index, correlation coefficient, performance index and the relative standard deviation of all simulated tests.

Table 2: Statistical validation parameters of each simulated test.

Test	***d***	***r***	***i***	***RSD* (%)**
40.1	94.55	92.07	87.05	3.41
40.2	93.62	94.75	88.71	4.53
40.3	96.48	95.12	91.77	3.36
50.1	97.01	95.35	92.49	5.79
50.2	97.44	95.93	93.48	4.16
50.3	97.38	96.33	93.81	4.18
60.1	97.62	96.63	94.33	5.24
60.2	97.53	97.92	95.51	6.66
60.3	98.37	97.77	96.18	5.21

All simulated tests of the proposed model presented values of relative standard deviation lower than 10 %, indicating satisfactory representation of the studied phenomena (Chen and Morey, 1989; Madamba et al., 1996; Mohapatra and Rao, 2005).

Moreover, all simulated tests presented values of performance index were higher than 85 %, indicating accuracy and precision of the proposed model to describe the system (Camargo and Sentelhas 1997).

Based on the analyzed statistical parameters the proposed model can safely describe the fixed-bed system to drying coffee berries, for the range of analyzed conditions.

3.3. Model comparison

Figure 5 shows the experimental observed values and MSU predicted results of moisture content for tests 40.3, 50.3 and 60.3.

The results showed in Figure 5 corroborate with the affirmation of Brooker et al. (1992) that the MSU model tends to overestimate the drying results, due to the considerations inherent to the model.

Table 3 shows the agreement index, correlation coefficient, performance index and the relative standard deviation of all simulated tests using MSU model.

Almost all condition simulated using the MSU model presented values of relative standard deviation higher than 10 % and/or a performance index lower than 85 %, indicating that this model not satisfactory describes the studied phenomena.

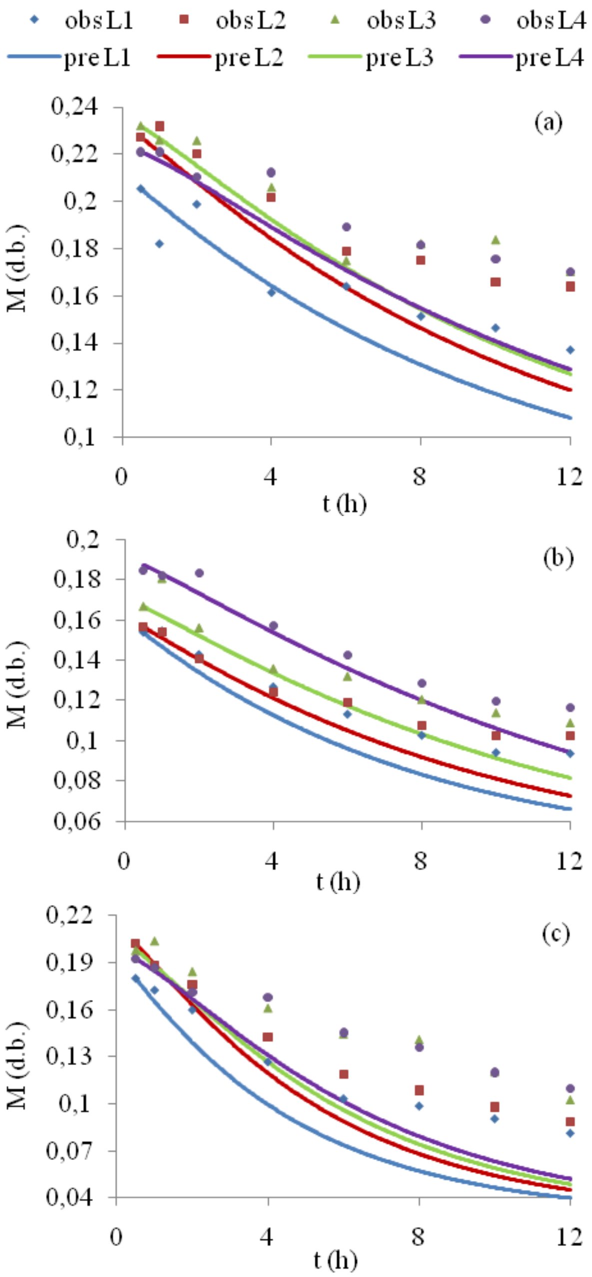

Figure 5: Observed and MSU predicted values of moisture content of test 40.3 (a), 50.3 (b) and 60.3 (c).

Table 3: Statistical validation parameters of each MSU simulated test.

Test	***d***	***r***	***i***	***RSD* (%)**
40.1	86.72	92.96	80.62	7.96
40.2	92.8	94.23	87.45	5.99
40.3	84.85	93.67	79.48	8.41
50.1	79.04	94.66	74.82	20.91
50.2	87.75	95.14	83.48	12.86
50.3	92.78	98.33	91.23	7.95
60.1	84.04	96.85	81.39	21.58
60.2	85.55	94.92	81.21	19.78
60.3	81.35	88.14	71.7	26.83

It can be observed by comparing Table 2 and Table 3 that all values *i* and *RSD* of the proposed model are better than the MSU model, indicating that the proposed model has a better performance.

3.4. Sensitivity Analysis

Table 4 shows all sensitivity coefficients calculated for each analyzed parameter, organized in increasing order of the S_t value.

Table 4: Sensitivity coefficients of each analyzed parameter of the proposed model.

S_i	S_M (%)	S_{rh} (%)	S_{Tp} (%)	S_T (%)	S_t (%)
T_s	119.11	170.24	75.93	85.03	115.16
M_i	91.26	46.80	27.65	50.43	46.30
T_{am}	29.91	95.58	41.98	34.82	45.30
V	53.37	44.12	22.94	31.69	34.71
rh_{am}	31.04	82.17	7.17	13.76	31.19
ρ_{ar}	30.67	46.81	22.61	28.56	29.87
L_i	27.58	42.77	27.17	30.97	28.85
p_{am}	23.34	33.78	20.66	24.01	24.32
c_{ar}	19.90	29.98	24.15	25.74	22.85
k	34.30	14.83	6.56	9.51	15.38
M_e	32.19	11.38	5.69	8.04	13.51
h_{lv}	14.15	18.98	6.50	9.83	12.73
h_{fg}	5.29	9.98	17.26	15.21	9.56
k_h	6.69	7.22	11.98	12.80	7.58
d_p	0.74	0.80	1.37	1.46	0.85
c_p	0.35	0.68	1.15	1.02	0.64
c_v	0.23	0.32	0.11	0.17	0.21
ε	0.05	0.08	0.01	0.03	0.04
ρ_p	0.01	0.02	0.01	0.01	0.01

It is observed in Table 3 that all input and initial parameters presented high sensitivity coefficients, higher than 24 %. The drying temperature (T_s) presented the greatest sensitivity coefficient (higher than 100 %), probably due to the high interference on the mass and heat transfer parameters and air properties.

The mass and heat transfer parameters and air properties presented substantial sensitivity coefficients, higher than 7.5 %. The air properties are only dependent of the thermodynamic conditions. They are well known and described on literature and can't be changed in the model structure. The transfer parameters are dependent of the fluid and particle properties, since the fluid is the air, the product transfer kinetics, represented by specific empiric equations, are the unique reason on changes of the model results.

All product physical properties presented low value sensitivity coefficients, less than 1 %. The variation of these parameters can be neglected, so an average or initial value can be used in order to simplify the model.

The unique product property that presented a considerable sensitivity was the water latent heat inside the product (h_{fg}). Probably, the consideration of this parameter on the model is one reason for a better performance than MSU model.

Furthermore, other reasons that can explain the improved performance are the consideration of pressure variation and layer volume shrinkage. As can be seen in Table 4, the model is very sensitive to the variation of L_i and p_{am}, presented a S_M value of 27.58 % and 23.34 %, respectively.

On the other hand, physical properties variation does not contribute to performance improvement of the proposed model, since the model is not very sensitive to these parameters.

4. CONCLUSIONS

From this research, it can be concluded that the proposed model can accurately describe the fixed-bed drying system to dry coffee berries. Furthermore, this model had better performance than the MSU model. Probably this result was due to the consideration of water latent heat inside the product, pressure variation and layer volume shrinkage. However, future studies must be completed to investigate the use of the proposed model to describe other drying systems for other products.

ACKNOWLEDGMENTS

The authors thank to CNPq and CAPES for the financial support essential to complete this work.

REFERENCES

Afonso Júnior, P. C., 2001. *Aspectos físicos, fisiológicos e de qualidade do café em função da secagem e do armazenamento*. Thesis (PhD), Viçosa Federal University.

Brooker, D. B., Bakker-Arkema, F. W., Hall, C. W. 1992. *Drying and storage of grains and oilseeds*. Westport: The AVI Publishing Company.

Camargo, A. P., Sentelhas, P. C. 1997. Avaliação do desempenho de diferentes métodos de estimativa da evapotranspiração potencial no estado de São Paulo, Brasil. *Revista Brasileira de Agrometeorologia* 5(1):89-97.

Chen, C.; Morey, R. V., 1989. Comparison of four ERH/EMC equations. *Transactions of ASAE*, 32:983-990.

Clarke, R. J., Macrae, R. 1987. *Coffee Technology*, vol 2. Elsevier: London. 321p.

Corrêa, P. C., Goneli, A. L. D., Afonso Júnior, P. C., Oliveira, G. H. H., Valente, D. S. M. 2010. Moisture sorption isotherms and isosteric heat of sorption of coffee in different processing levels. *International Journal of Food Science & Technology* 45:2016-2022.

Dalpasquale, V. A., Sperandio, D., Silva, L. H. M., Kolling, E. 2008. Fixed-bed drying simulation of agricultural products using a new backward finite difference scheme. *Applied Mathematics and Computation* 200:590–595.

Dincer, I. 1998. Moisture loss from wood products during drying - Part I: moisture diffusivities and moisture transfer coefficients. *Energy Sources* 20:531–539.

Fritzson, P. 2003. *Principles of Object-Oriented Modeling and Simulation with Modelica 2.1*. New York: Wiley-IEEE Press.

Guiné, R. P. F., Rodrigues, A. E., Figueiredo, M. M. 2007. Modeling and simulation of pear drying. *Applied Mathematics and Computation* 192:69–77.

Hamby, D. M. 1994. A review of techniques for parameter sensitivity analysis of environmental models. *Environmental Monitoring and Assessment* 32: 135-154.

Henderson, S. M., Perry, R. L., Young, J. H. 1997. *Principles of Process Engineering*. Fourth edition. St. Joseph: ASAE Press.

Izadifar, M., Mowla, D., 2003. Simulation of a cross-flow continuous fluidized bed dryer for paddy rice. *Journal of Food Engineering* 58:325–329.

Lecorvaisier, E., Darche, S., Silva, Z. E., Silva, C. K. F. 2010. Theoretical model of a drying system including turbulence aspects. *Journal of Food Engineering* 96:365–373.

Madamba, P. S.; Driscoll, R. H.; Buckle, K. A., 1996. Thin-layer drying characteristics of garlic slices. *Journal of Food Engineering*, 29:75-97.

Mohapatra, D.; Rao, P.S., 2005. A thin layer drying model of parboiled wheat. *Journal of Food Engineering* 66:513-518.

Sfredo, M. A., Finzer, J. R. D., Limaverde, J. R. 2005. Heat and mass transfer in coffee berries drying. *Journal of Food Engineering* 70:15-25.

Vossen, H. A. M., 1979. Methods of preserving the viability of coffee seed in storage. *Seed Science and Technology*, 7(1):65-74.

Willmott, C. J., Ackleson, S. G., Davis, R. E., Feddema, J. J., Klink, K. M., Legates, D. R., O'Donnell, J., Rowe, C. M. 1985. Statistics for the evaluation and comparison of models. *Journal of Geophysical Research* 90(5):8995-9005.

AUTHORS BIOGRAPHY

Emílio de Souza Santos, Agricultural and Environmental Engineer, graduating in Agricultural Engineering (Universidade Federal de Viçosa). Regional Development Analyst at CODEVASF (Companhia de Desenvolvimento do Vale do São Francisco e Parnaíba). *Work Areas*: Operation, Maintenance and Safety of Hydraulic Infrastructures; Water and Environmental Management.

Paulo Cesar Corrêa, Agronomy Engineer, Ph.D. in Agricultural Engineering (Universidad Politécnica de Madrid). Associate Professor of Universidade Federal de Viçosa. Instructor and Researcher at CENTREINAR (Brazilian Grain Storage Training Center). Editor of Brazilian Journal of Stored Agricultural Products. *Research Areas*: Physical Properties of Agricultural Products; Storage and Process of Agricultural Products.

Brian Lynn Steward, Electrical Engineer, Ph.D. in Agricultural Engineering (University of Illinois). Associate Professor of Iowa State University. Member of ASEE (American Society for Engineering Education) and ASABE (American Society of Agricultural and Biological Engineers). *Research Areas*: Virtual Prototyping; Dynamics Systems Modeling and Simulation (off-road vehicles and agricultural systems); Machines and Agricultural Automation.

Daniel Marcal de Queiroz, Agricultural Engineer, Ph.D. in Agricultural and Biosystems Engineering (Purdue University), Associate Professor of Universidade Federal de Viçosa, Director of CENTREINAR (Brazilian Grain Storage Training Center); President of the Arthur Bernardes Foundation in; President of the Brazilian Society of Agricultural Engineering. *Research Areas*: Machine Design; Precision Agriculture; Grain Drying.