Dr.-Ing. Björn Döbel
Münchner Straße 42
01187 Dresden

Univerzita Karlova v Praze
Dekanat Matematicko-Fyzikalni Fakulty
Studijni oddeleni - Doktorske Studium
Praha 2, Nove Mesto
Ke Karlovu 3
PSC 121 16

Dresden, August 11 2015

To whom it may concern,

I have read and hereby review the Doctoral Thesis submitted by Mgr. Martin Děcký titled „Application of Software Components in Operating System Design." In this thesis, Mr. Děcký presents the design of the HelenOS microkernel-based operating system. The thesis focuses especially on designing an operating system based on component-based software engineering principles and on development processes to support the collaborative development of a dynamic general-purpose operating system.

This focus on the development and design process can be considered a novel contribution of Mr Děcký's work that augments other operating system designs, which in contrast to this work traditionally concentrate on implementation aspects (performance, security, reliability). It is this why I consider this work a suitable demonstration of the author's ability for scientific work and a summary of a huge effort in terms of operating systems engineering (even if this effort is not only the author's). I therefore recommend that the committee accepts this doctoral thesis.

In the remainder of this review I am first going to shortly summarize my main takeaways from this work before I thereafter outline points of criticism that should be addressed in the defense or in future work.

## Contributions

The thesis is motivated with the need for an operating system to be used in research and education at the authors' university. Special focus is put on making the operating system reliable, that is avoid instabilities in the system right from the start. A second goal is making the system practically useful, that is supporting a wide range of functionality while being able to reach this goal given the resource constraints the team faces. These two goals motivate a focus on development methodology instead of implementation detail.

The initial motivation is followed by Chapter 5 discussing related work, especially how other operating systems influenced HelenOS' development. This part of the thesis covers all relevant existing systems and explains how these works differ from what HelenOS provides. The author then discusses component-based software engineering and comes up with a set of nine design principles for an operating system in line with the thesis' goals.

The design principles are then applied in a rather short discussion of HelenOS' kernel features (Chapter 6) and a rather lengthy look into the development process (Chapter 7). In parts, this discussion of the development process reads a bit too philosophical for my taste as it lacks technical detail. On the other hand this is with no doubt a reasonable approach because of the chosen focus on the development process and the fact that people are a major contributing factor to any software.

The thesis becomes more technical in Chapter 8, when Mr. Děcký describes the various approaches HelenOS takes to ensure that as few bugs as possible end up in code. He discusses different approaches of verifying correctness (formal verification, model checking, regression/unit testing, static analysis) and how existing tools are leveraged to make HelenOS more reliable. Instead of building everything from scratch (a common problem in research), the system is built by "standing on the shoulders of giants" here, which makes this chapter a commendable contribution to reliable system design.

## Major criticism: Lack of performance evaluation

While the thesis' focus lies in development methodology, the end product is a computer operating system and the thesis claims that HelenOS reaches the given design goals. While this is acceptable from a qualitative point of view, the thesis completely lacks a traditional evaluation of quantitative properties. It is a fundamental principle of operating systems design to evaluate the results of one's work using standard benchmark programs and compare those results to other systems. This thesis contains none of this at all. Even if a system does not provide standard interfaces, we must have some way of understanding performance in order to assess whether HelenOS is actually useful.

Let me give three examples:

1. When discussing HelenOS design principles and kernel features, Mr. Děcký argues that traditional microkernel systems strictly avoid implementing any complex data structures or algorithms within the kernel and require those to be moved to user-level components. He argues that HelenOS forgoes this strictness and uses complex features, such as concurrent hash tables to improve kernel resource management.

   Unfortunately, this argument is made purely from a philosophical standpoint. The argument would have been much more convincing if it was backed with some kind of benchmark that shows the improvements such complex algorithms provide. Reading the presentation in the thesis and putting myself into the "advocatus diaboli" role and putting a lot of irony into the following statement, I would condense the statement into "In order to properly motivate the complex validation steps discussed in this thesis, we first had to put some non-toy algorithms into the kernel, because otherwise we wouldn't have needed this validation at all."

2. Moving away from pure performance evaluation, the thesis also does a poor job in quantitatively analyzing the system's reliability. Mr. Děcký duly notes that each of the validation activities incorporated into HelenOS build process was able to pinpoint bugs that might have otherwise gone unnoticed. Unfortunately, this usually only mentions a number of change sets to the HelenOS system, but does not give details on the number of bugs found in the system and how these numbers relate to other operating systems.

I admit that such an evaluation is much harder to do than running a plain benchmark or given a number of found bugs. However, this analysis would help strengthen the argument for using as many analysis/testing tools as possible.

3. As I already mentioned, I liked the way using external tools for improving reliability throughout the build process was presented in Chapter 8. However, using those tools comes at a price in terms of development effort (someone has to add the respective annotations etc. to code), compile time (how long do those tools run?), and perhaps runtime performance. The thesis unfortunately does not quantify these costs in order to relate them to other approaches.

All in all, my main criticism of the presented thesis is the lack of hard numbers in order to understand the tradeoffs HelenOS' authors chose. The thesis would have much benefited from a less philosophical evaluation that allows statements such as "We did A and this cost us X% in development effort and Y% in performance, but in turn we were able to increase reliability / practicality of the system by Z%."

## Moderate criticism: Partial lack of technical detail

Coming from an operating systems background, large parts of the thesis were too philosophical from my perspective and I seriously missed discussion of technical detail. Again, let me give two examples:

1. Chapter 6 gives a very brief (4 page) tour of HelenOS' kernel features, while Chapter 7 spends 12 pages discussing HelenOS' development process.. Being an operating systems person I would have enjoyed the ratio to be the other way round, but I accept that Mr. Děcký chose to put his focus on development methodology instead.

2. Section 8.3.3.1 spends less than half a page on describing a byte-code interpretation mechanism that allows the kernel to execute user-provided device driver code in kernel mode. While the idea of downloading user code into the kernel is not new (exokernels had that back in the 1990s), the idea of protecting the kernel from this unsafe code by using a byte code interpreter sounds pretty novel and in the position of the author I would have put much more focus on this topic (the thesis provides few detail on the implementation and does not even start to discuss cost vs. benefit).

## Minor criticism: Lack of citations

Scientific style demands using citations to allow the reader to get pointers to other work that is discussed in the context of a scientific publication. While Mr. Děcký in general adheres to common standards, I would have preferred a much higher rate and amount of citations. Especially, when talking about related work from the operating systems area, Mr. Děcký names lots of projects (Plan9[1], Mach[2], Genode[3], Barrelfish[4], Nova[5]) without giving

1 Link to http://plan9.bell-labs.com/plan9/ would have been nice.
2 Richard Rashid, Daniel Julin, Douglas Orr, Richard Sanzi, Robert Baron, Alessandro Forin, David Golub, Michael Jones. **Mach: A System Software kernel,** COMPCON 1989
3 Norman Feske, Christian Helmuth: **Design of the Bastei OS Architecture,** TU Dresden 2006
4 Andrew Baumann, Paul Barham, Pierre-Evariste Dagand, Tim Harris, Rebecca Isaacs, Simon Peter, Timothy Roscoe, Adrian Schüpbach, and Akhilesh Singhania. **The Multikernel: A new OS architecture for scalable multicore systems**
5 U. Steinberg, B. Kauer **NOVA: A Microhypervisor-Based Secure Virtualization Architecture**, EuroSys 2010

any pointers to their sources, be it scientific publications or rather web pages. The same applies to sections where criticism of microkernels is countered but no reference to any such criticism[6] is provided.

[illegible]

Signed: Björn Döbel, Dresden, August 11 2015

6 Such as Steven Hand, Andrew Warfield, Keir Fraser, Evangelos Kotsovinos, Dan Magenheimer: **Are Virtual Machine Monitors Microkernels Done Right?** HotOS 2005