PhD Progress Report 1

Lorand Kedves
2016/2017. II.

Structural Transparency, Implementation and Runtime Optimization in Software Engineering - Theory and Practice

Lorand KEDVES

Doctoral School of Information Science and Technology
University of Pannonia
Supervisor: Dr. Botond BERTÓK

Foreword

As a father of three secondary school sons and correspondent doctoral school student, I had to adapt my schedule to my actual work tasks, which were plenty and allowed much less academic activity than assumed. However, they also offered experience strongly related to my doctoral research, which I will list here (hopefully without breaking confidentiality).

Experience from Work

Efficient Coding in Java

Situation: Our task is to replace a 15 years old complex form editor application with a new Java version with a short deadline. The heart of the application is a VYSIWYG form designer with direct connection to other layers (validation, error correction, calculated attributes, etc.) and exotic features like global undo. The project has speed, coherency and stability issues. This is the practice of my core question: *how software represents our knowledge?*

Solution: After a thorough analysis and prototyping, the graphics subsystem was replaced with a new implementation with features related to my research, some of which I list below.

1. **Data objects** - "In Java, everything is an object", a common statement with serious consequences, like tons of classes generated from database tables by Hibernate, without thinking of the system architecture. The problem is: we have no control on changes in the data layer, which results in yet other heavy-weight tools like reflection or aspect oriented bytecode manipulators. I have created a simple data storage mechanism based on EnumMaps, so data object types are not represented by specific classes, but enums listing their fields. The storage has multiple API layers (read-only flat and multivalue, and mutable). The result is a very flexible data storage layer, script-like features in Java, and total control over events: I know when any field is read or set.
2. **MVC** – Based on the Data storage features, I could create a clear MVC architecture with strict separation. Data related logic receives control automatically on any change so they can react on them; a completely separated View layer is notified on any change so the display is refreshed immediately, without any direct programmed connection to any other component; the same applies to the Property Editor.

3. **Undo** – contrary to the standard approach (creating and storing undo steps when doing the action), all changes appear at the bottom, in a master listener of the data layer; can be stored and played back.

Result: This approach earned management trust, we could solve critical problems of a year of development literally in weeks, created a framework that allows fast and reliable development now, and supports flexible adaptation to changes in the future.

Knowledge Card Management

Situation: As a consultant of an international pharmaceutical company, I participate in a project to survey an existing internal data and process management system, collect and organize knowledge, create evaluation criteria for the new system, and find an optimal solution. This is the other end of my research: how to enter raw human thoughts into an information system, how to help participants improve their cumulative knowledge.

Solution: I recommended to create a small "knowledge card manager" application. The core features of "how to think together with a machine":

1. **Knowledge Cards** – A5 pages containing a one-line title, tags, and plain text. The card must focus on one single statement, if it does not fit, it must be split. One interview, discussion, or personal analysis produce several cards. The knowledge remains granular, it is easier to review 5 cards after an interview than a long flow text. The system allows free text search and tagging, so it is manageable even with thousands of cards.
2. **Tagging** – you can assign any number of tags to the cards. Tags represent your opinion or organization of the core facts (that are the cards). This is the second level of thinking: list the possible values of an aspect, and flag the cards with them, so you can see how each kind is represented or supported by the exiting facts – so you can ask new questions, investigate missing areas, etc.
3. **Filter, compare, aggregate** – you can use tags to find the facts you want to work with; compare the understanding of a team by having them assign the same tag set to the facts and compare results. Assign numerical values (weights and scores) to tags, and you have a decision tree, which is still dynamic, allowing to do what-if analyses, and improve your weighting based on actual data.

Result: We are currently in the negotiation, nothing implemented yet, but we already use knowledge cards in Word files. I plan to implement this system as part of my doctoral study, even in official university cooperation in case they prefer so.

Academic Studies

OTDK Lecture

I prepared the lecture and participated at the OTDK in Veszprém with part of my thesis, "Software as a Data Structure". Unfortunately, the video file was broken, only the presentation ppt is available from the event.

The main critics was related to how much practical support and use there is behind my high goals. I tried to change my presentation according to this need as much as I could, but unfortunately, from all public and private responses it turned out that I failed on this task.

This feedback changed my approach to my doctoral study as well, I plan to focus on what I know: actual system design and development, and give a lower priority to communication. (This already affects the organization of this status report as well.) I see lots of practical tasks that I can work on or started already, which will result practical, visible experiences, or even usable frameworks. For publications, I should use these results, instead of trying to "sell the idea".

Prof. Alan Kay

I have always searched for scientific foundations of my research. This happened before, when I met my development habits formal descriptions in the GoF Design Patterns book, system analysis and documentation ideas in SPICE/CMMI software quality assurance standards, etc. During my MSc studies, I had the same feeling with finally understanding the importance of Turing machine analyses and proofs, including the Church-Turing theorem; the essence of the von Neumann principles, the mathematical background of entropy, compression and error-awareness, and their relation to actual software development.

When preparing to the OTDK lecture, I met Alan Kay's lectures. He was the leader of Palo Alto Research Centre when and where most of today's IT environment was envisioned and partially, implemented as the environment in 1960-70 allowed. I also started reading and watching Ivan Sutherland, Seymour Papert, plan to read more from Marvin Minsky. Their research areas, from Lisp programming, Smalltalk / Squeak environment, etc. are the true foundation of my vision about dynamic knowledge representation and human-machine interaction. It will be very interesting to understand their views and the reason why that much more efficient approach seems to fail in today's informatics environment.

Courses

Analysis and Synthesis of Technical Processes – Dr. Botond Bertók

We discussed over the basic usability of P-graphs in modeling and optimizing processes in industrial environments, and the importance of such achievements, taking Zoltan Süle's findings. I have studied some actual P-graphs, and compared to my software modeling methods, where the main difference is the lack of returning edges. In software, any function with a return value, or a simple access of the variable value is an edge going "out" just to "get" something back. I found this difference fundamental and considered that P-graph methodology should be extended with a better support for dealing with this edge class.

Further thinking however revealed that my approach is wrong: if I see the "process" from the other point of view, and put the available data into the focus, see the actual processing algorithms as nodes, and consider the execution state "outside" of the code block, that allows a different, and scientifically more transparent process representation method.

I started analyzing this (inherently networked) approach, ad so far it is more promising than any textual version could be. I also started using it, as follows.

IT Tools and Techniques for Process Optimisation – Dr. Botond Bertók

We are working on and optimized subset finding algorithm, which is related to ongoing researches. I started the implementation of an algorithm testing framework for that (available on GitHub). It naturally contains a thread-safe timing service that focuses on exact measurements, and exclude its own "housekeeping" as much as possible. I plan to extend it to multi-threaded environments. The more important element however is, utilizing the graphical representation, other "actions" that implement the 3 control structures (Sequence, Iteration, Selection); others integrate runtime environment features through Java reflection (class instantiation, function call, member field get and set), the current implementation supports "pipelining" actions.
I have implemented different versions of the algorithm, using simple Set operations, utilizing BitSet in optimized sorted fashion, but also only using BitSet without optimization to see how fast the JRE implementation is.

Later extension ideas: process multivalues, use thread pools, and instead of pipes ("unix programming"), I will extend the runtime context with identified data items, so the function call parameters can be provided. The target is to test the algorithm versions with various multi-threaded settings, only by changing the node configuration.

Bin Covering – Dr. György Dósa

In this course, I study process optimization literature and methods. The main goal is to get familiar with the patterns and research results, so when the above representation will get more elaborate, I can see, test, or even implement some automatisms to optimize the process graph, or adapt the actual execution parameters according to measurements.

We also talked about an existing publication plan built on configurable bin covering algorithms, where a "meta algorithm" can monitor the fitness result of the current settings, experiment with parameter modifications and by adapting to the actual input, achieve better results for an online input situation, where only the actual input is available for the algorithm, and requires immediate decisions.

Graph Coloring – Prof. Zsolt Tuza

Studying graph algorithms is also very interesting for my research: when I see the software as a fractal-like graph, such methods can help optimizing the structure, detect architecture weaknesses. A simple example is: with bytecode authoring tools, it is possible to create a dependency graph between modules, packages or even classes. We can then color various modules and see where they appear in the codebase. Critical modules in a Java system can be:

- Reflection, because that allows touching other segments behind the scene, without direct source reference;
- GUI (for example, Swing): the more dependent a system is on one GUI layer, it is harder to port or create a console or server-side version;
- Data source items (XML, Database, JSON, etc.) objects: the application is more sensitive to external storage or syntax change;
- Any external toolkit module: the system is sensitive to changes in them, which makes it harder to switch to newer versions;
- …

We also discussed implementing graph algorithms to be used for education.

I also encountered a task in the form designer, where I created a new algorithm for splitting screen elements to separated regions. I used interval graphs to find not covered sections to split the area. It runs in linear time, less then 10msec for random source (2000x1000 area, 100 random blocks). I extended the original solution to "n" dimension, where the only requirement is that the data representing endpoints is "comparable" with each other.

Budapest, 2017.05.22

Lorand KEDVES
PhD Student

Dr. Botond BERTÓK
Supervisor