



Science meets business

Collection of challenging topics for ambitious students

Range of Topics

| | |
|--|-------|
| Analysis of web systems and observation of their run-time behavior | I-VI |
| Analysis of user behavior in heavily populated systems | VII-X |
| Organization development | XI |

For your

(Internship)
Bachelor Thesis
Master Thesis
Dissertation

Analysis of web systems and observation of their run-time behavior

... and possible optimizations

Even properly tested software tends to "misbehave" from time to time. This might be caused by high traffic, a particular configuration of input data, partial system failure or things of that nature. Misbehavior might result in things like corrupt data, helpdesk tickets, total processing failure ("internal server error"), partial system failure and of course unsatisfied customers. We don't want that, but it happens.

When things like that occur, it is usually very hard to trace them back to their cause if you don't observe the running system properly. Some of the things - like a full disk - can even be prevented when monitoring the system. Of course, we test our software and monitor our systems properly, but we can do even better. This brings us to some non-trivial problems (We think ^), which we will describe hereafter.

I) Traceability of run-time behavior across system boundaries

Many times requests, HTTP requests for instance, are processed in different subsystems, like a reverse proxy, a web cache, an application, some internal services, etc.. Also many requests result in responses to the client, which in turn makes several other requests that are logically attached to the initial request (AJAX). We want to be able to observe the trace of processing and to group requests. Since most web applications are parallel in nature, grouping by time is not an option.

One solution idea is "The Request Pulse", which is described on <http://devblog.icans-gmbh.com/passion-the-request-pulse/>.

II) Prediction of workload of system resources => computer-aided resource planning

Although virtualization and cloud computing are great instruments to react quickly on changing requirements, adding a new instance to the pool may take some time or might be risky, depending on the technology you're using.

Some day we might be forced to scale out a MySQL database by sharding or replication, which will take some time for preparation and the actual migration. Since we don't want to be forced to rush things, we'd like to be able to predict when we have to react to resource shortages of that nature.

Traditional monitoring solutions like Nagios or Zabbix are great, but we want to reduce the manual effort even more so that we can scale without worries.

III) Static and dynamic software analysis with regards to dependencies within data and data manipulation

This might be a pretty tough one. The goal is to be able to trace a piece or class of data back to its origin. For instance, we want to be able to trace a cache, which is usually composed of different pieces of data, back to its origins, so we can in turn recalculate that cache if a part of it changes.

Another possible application is to be able to trace back the origins of a statistics, which is usually composed of aggregations or "interpretations" of data. So we could determine which data processors would be influenced if we changed the value range of X. Having that option would be great, because we need to act fast within large systems, without negative impact on business decisions or live analytical applications. It would also provide new ways to dynamically generate documentation and verify that the implemented systems actually follow their specifications.

IV) Benchmarking of new features in our production environment before their actual release

When determining the performance hit of a new feature we face two problems:

- a) Tests are good, life is different
- b) A mirrored test setup tends to get very expensive

This is why we want to test the performance of a new feature without actually releasing it and without noticeable disturbing the processing of the users request.

V) Domain-specific health reporting

We use Zabbix to monitor our servers and infrastructure. A software component should be able to diagnose its own health status and should be able to report its "own opinion" of the situation to the monitoring system in addition to the standard values, which are being observed by Zabbix. Things the component might take into account could be workload, unit tests, functional tests, health reports of its children, etc.. An additional evaluation method might be to make the component learn what's "normal" and to react on deviations.

VI) Pre-emptive page preparation

Assume there is a website, which redirects the visitor to his/her personalized home page after he/she authenticated himself/herself. This home page contains personalized recommendations of articles, the latest ten threads of all the boards the user "watches" and his inbox of private messages. The retrieval of all needed data used in those little widgets naturally takes some time. If we can't speed up the retrieval itself, we could try to do it before the actual request has been sent.

In our simple example, we know "if the visitor successfully authenticated himself/herself, we will have to retrieve his recommendations, latest threads and the inbox". So we could trigger those preparations well before we need them. For example when the user submits the login form, just types his user name or even just when he/she re-visits the website and we can identify him/her (e.g. by cookie). We could also try to predict which pages a user might watch next by analyzing his/her behavior or the behavior of many visitors. Since we sometimes only assume the identity of the user - who hasn't authenticated himself/herself at that point - we have to take some security measures to prevent abuse of this system, of course.

This problem is closely related to the following:

Analysis of user behavior in heavily populated systems

VII) Process Modelling for Online User Behaviour

Using the web has become a substantial everyday activity for many people. Due to the massive amount of available content, users adapt to the general web-browsing experience and develop behavioral strategies for interacting with the web and from which expectations regarding the usage of unknown websites or applications are deduced. These implicit behavioral strategies play a crucial role in topics like the design of user interactions, but also provide a way to identify the level of web experience of a user by analyzing his online behavior. We want to model the process which is used by people to generally interact with websites to improve the customer experience on our websites, but also want to investigate the potential of using such a behavioral model to tailor websites to match the level of experience of the user. Other interesting topics would also benefit from such a model: For one, could it be used to predict the most likely pages a user will visit next? These pages could then be rendered preemptively to provide much faster delivery or would allow the computation of customized results in near real-time. A second interesting question is if such a model is able to identify a certain web client as being operated by a real, interested human being or whether it might be an automatic program or a person who is paid to follow a certain routine. This would have various applications in online fraud prevention and can additionally aid in customizing the delivered content as well.

VIII) Measuring Online User Behaviour: Metrics and Methods

The web today already allows to embed rich applications in websites and thus to create an increasingly interactive user experience. Customizing this user experience poses the challenge to react on the user behaviour in near real-time which requires knowing on which triggers to act upon. We are thus interested in creating a catalog of possible metrics that can be measured in regards to online user behavior, identify methods by which such metrics can be acquired and to connect these metrics to possible use cases. Such use cases can be found in various applications like interactive avatars, customization of delivered content, online fraud prevention, optimizing marketing strategies and tuning the software and hardware infrastructure to optimally handle the demands of the users.

IX) Simulation of the Dynamics of a Sequence of Poker Games

Poker is a challenging game, it requires skill and this skill can be developed. As such it gained much recognition in the online world in the last years, which is likely to increase due to the current efforts in the EU to create a safe, legal and regulated poker market. If played fair, poker can be a rewarding game to learn, but there are also many fraudulent attempts to acquire an unfair advantage in the game, for example by cooperating with peers in competition to further unsuspecting players. In order to spot sophisticated fraudulent behavior, we want to increase our understanding of the dynamics that develop while playing poker over time. To do so, we want to simulate a series of poker games and track certain metrics over time, either by using real poker history data or by modelling certain circumstances statistically and then studying the effects of the temporal dynamics and the general outcome of the game.

X) Large-Scale Graph Mining to calculate Metrics in Online Poker

Poker is a game composed of a series of transactions. These interactions can be modelled as a property graph and we are interested in identifying metrics that help us to prevent or identify ongoing fraud in online poker. To this end we want to mine massive amounts of history data of real poker games stored in a distributed Hadoop environment using the Pregel model developed by Google for large-scale graph processing. Tasks therefore include understanding the dynamics of the poker environment; suggest interesting metrics to look at, implementing Pregel-like algorithms to compute such metrics using an existing Pregel-like framework in Java and analyzing the final results.

Organization development

XI) Continuous Decentralized Appraisal System

We at ICANS believe that we can only deliver high performance continuously, by working in a passionate team of very smart, skilled and result-oriented individuals. We believe that continuous and highly qualified feedback is an important key to continuous development. We believe that peer to peer feedback is at least as important as feedback from the management.

Therefore we want to create a simple to use system, encouraging everyone in our organization to give performance based feedback to peers, teams and managers systematically on a monthly, weekly, or even daily base. Doing so will enable the organization, the management and each employee to take care even more of the personnel progress in near-real-time and a highly scalable manner.

To successfully design and implement such a system, we have to pay attention to psychological, functional and technical aspects, equally.

Final words

These are some of the things we haven't got the time to tackle, yet. We don't have solution ideas for some of the problems and we think, some of those might be really challenging. We also hope there is scientific value in many of the topics. Some of the solutions would be really nice to have, others would be awesome. Perhaps some problems aren't solvable at all, under the given circumstances. Hopefully many of the solutions will result in open-source projects.

Interested in mastering these or maybe other challenges together with us? Let us know!

Call us: **+49 40 22638290**; Write us: bewerbung@icans-gmbh.com; Visit us www.icans-gmbh.com.
For more details also see: <http://devblog.icans-gmbh.com/stop-bis-hier-hin-hast-du-alles-richtig-gemacht-studium-wissenschaft-bei-icans/>