Post mortem ICPC 2012

From ICPC-Contest Control Standard
Jump to navigationJump to search

What went well:

- We got access to the problem set and the test data about 2 hours before the contest and Per Austrin guided us through the problems which was great.
- Assigning the problems to specific analysts who also presented them in the broadcast. Having two people watching the top 10 teams and one person communicating with the studio was too many, one person can do all of that.
- Our tools (Kattalyzer, Codealyzer, iCAT, twitter export, KATTIS and DOMjudge) worked very well and gave us much better information than last year. The activity graph, especially for teams, was a gold mine.
- The new analysts were great! Now we had enough people to really cover the contest and present all the problems.
- Everything was ready for the dress rehearsal so everything was running as if it was the real contest. This was to a large degree due to the world finals test weekend.
- The automatic recording of the webcam when top teams submitted a problem worked really well and was fun to watch.
- Presenting three problems every our hour worked well.

What can be improved:

- The interaction with the studio was not tested before the broadcast started.
- The wired network did not work properly until Monday afternoon which delayed our work.
- The wireless network did not work properly at all, even though there were several available networks.
- The interaction with the production can be improved, when we had videos and graphs to show we had to deliver them on a USB stick even though we had set up a NFS share with the information.
- The computer assignment was a bit unclear, mainly due to the fact that we became twice as many as originally intended.
- We would like a printed copy of the problems for each analyst.
- The IRC channel had almost no activity, better interaction with the viewers would be nice. ICPCStudio had a separate twitter account, but questions there were nearly all "can you show $myteam?"
- Publishing the problem set was ad hoc and took a while. Someone should have the task of doing that first-thing in the contest, because there's a lot of demand for it.
- Contest feed had to be restarted, and this caused some manual work. We should ensure that rejudged problems are handled properly, by the contest control system as well as all of our tools.
- Twitter feed worked. But unfortunately not too many seem to have known about it.

What are important things to do for next year:

- Develop a ticker with breaking news that can be shown even during interviews and pre recorded segments. Develop a separate iCAT feed for this.

- Get annotated test cases from the judges to provide even more detailed automated information to the studio.
- Get support from the official CCS to run all test cases even if some fail. (Jaap: I can imagine that sysops/Kattis doesn't like this. Furthermore, we did set this in DOMjudge; did this not work as hoped?)
- Import all the background data directly from the registration system without manual work.
- Prepare the list of expected top teams well in advance, not on the morning of the contest. More information about teams like who are expected and unexpected.
- It would be very interesting to make a public version of the iCAT interface since it would allow the coaches to follow their teams in detail.
- (Semi-)automatically create a queue of interesting submissions to be manually inspected by an analyst (based on rank and potentially judge output), this might best be integrated in an automated judge. If new submissions arrive for the same problem only the latest submission is relevant.
- Automate as much as possible of the work we did manually:
    - Adding events when teams start working on a new problem (need to handle teams with template code)
    - Give failed submissions for top teams a higher priority since this information is very interesting
- Have one configuration file for iCAT and related software that covers all static data relevant to a particular contest (contest start time, balloon colors, problem names, number of problems, possibly team information, etc.). By the end of the actual WF we had config information in several places despite trying to have one file. This was made more difficult because we had many different languages: PHP, Javascript, Java, Python, etc. Perhaps a JSON or YAML configuration file would be the best format.
- Have one notion of time to unify events that occur during a contest. I (Greg) think it should be wall-clock time UTC, because that is always unambiguous (as opposed to contest time, which repeats itself each time we hold a contest).
- iCAT
    - Redesign the iCAT database
    - Support restarting the feed
    - Make a flexible interface in iCAT to refer to an external judging system for more information.
    - Automatically fill in the contest time and the analyst name in the iCAT interface.
    - Include a clock in the iCAT interface.
    - Show the age of each entry rather than its contest time
    - Support to change inconsistent and incorrect data
    - The feed viewer in iCAT is somewhat restricted in the types of events it can display -- i.e. each event is expected to be a single row from a single table; it would be nice to be more flexible to be able to join tables. However, we have to balance flexibility with security of the database (SQL queries sent over http can be a vulnerability).
    - Add a way to view the entire directory structure of a team from iCAT?
    - A dedicated communication channel between the studio and the analysts
    - Include the hypothetical scoreboard in the team view
    - Include small test cases that team often fail on include statistics on how many teams have failed on this
- Codealyzer
    - Check how many teams have good test cases, check if they are correct
    - Compute average coding time

    - Compute average order of problems, does this separate top10 from top20?
    - Compute average number of parallel active problems
    - Have a web-based interface for creating Codalyzer rules, evaluating them, adding them to the active set of rules, and discovering conflicting rules.
    - Allow Codalyzer to detect editing events, but perhaps reserve judgment on which problem each file goes with until display time. This way we can refine our idea of which edits go with which problems on the fly, including edits that occurred in the past.
- Larger blackboard with colored chalks
- Prepare complete images to computers (Jaap: I suggested this, but maybe this is not feasible in an unknown environment; alternative: document a list of system setup actions)
- More integration to watch team files in a single interface
- More structural feeding of content to others than studio (twitter, irc, ...). We did post a small and rather random selection of things to these channels. Does studio want some kind of 'exclusive' or 'first right'?
- The idea of recording teams that submit solutions that - if correct - will get among the top n teams is great, and can definitely be developed further. This year, we had no simple way of sending a good clip to the video production, and they had no obvious way either of showing it.
- More graphs or interesting stuff to show to the studio. Perhaps a way of providing a screenshot/graph along with a summary on what it shows. And some measurement on how interesting we think it is. If we prepare a queue of, say, 8 graphs, we give the people in the studio something to talk about whenever there is an awkward delay... For instance when waiting for the scoreboard resolution to be fixed.
- Remember to backup all data after the contest
- Automatically produce a video for every team with its first 20 seconds of the contest, every solved problem with a scoreboard which shows the change, and then the last 20 seconds of the contest.
- More detail in the information added by analysts in iCAT.
- In depth spy reports from the top teams what they are working on, whether they are on the right track, how they work in the team and so on.
- Each problem presentation in the studio should have the format: describe the problem and the solution briefly, describe special cases that the teams often fail on.

Retrieved from "https://clics.ecs.baylor.edu/index.php?title=Post_mortem_ICPC_2012&oldid=1646"

- This page was last edited on 12 June 2012, at 19:23.
-