Feedback Manual

Feedback Session (1st Simulation RUN)

20.11.2006

General:
In order to provide suitable feedback during an AMEISE course, different effects of the QA-200 model have to be considered. This document suggests a simple step-by-step guide for tutors of AMEISE simulations. It explains how to prepare an AMEISE presentation and how to interpret the generated handout.

Prerequisites: Handouts generated by "Aorta"

I. Step-by-Step Guide for an AMEISE presentation:

1. Start the "Overall Feedback" presentation
    a. It should contain enough motivation for students (→ Congratulation)
    b. Comparison of results
        i. Good idea: USE COLORS
        ii. There is NO WINNER/NO LOOSER
        First Game: It is important to learn about effects, not to win!
        iii. Only imperfect simulation runs can be used to demonstrate effects of decisions

2. You, as a teacher, decide: What is the FOCUS of the presentation (→ time constrains). At least the following two aspects have proven to be instructive in practice:
    a. Focus on cost
        i. Tell the students, why there are differences in COST
        For this reasons: select 2 or 3 games of nearly the same duration (days) but different cost!
        ii. Show GANTT charts of the 2 or 3 simulations
        iii. Show the number of developers of each phase, talk about communication overhead
        iv. Show the ratio between cost and months
    b. Focus on quality
        i. Select 2-3 games with interesting number of AFPs and # of errors
        Then: answer the question: WHY is the number so different?
        ii. Best way: Explain the role of the customer, the effect of the qualification of the review-team
        iii. Higher Quality? Answer: 2 Reviews!!!!
            1. Best person for specification!
            2. One review with customer
            3. The author should correct the specification
            4. Another review (with or without customer)
            5. The author should correct the specification
            6. Errors found in later phases: "CORRECT ALL DOCUMENTS"

3. Present some observations of different simulation runs
    a. Dependent on the FOCUS of the presentation. However,
        i. (Nearly) every game: Problems with productivity
        ii. Problems with quality of specification → design → code
        iii. Problems with review (without customer)
        iv. Problems with review/correction process
    b. Demonstrate propagation of errors
4. Give some final guidelines for second game
    a. Very important: Do not optimize everything at once!
    b. Better: 2nd and 3rd simulation
    c. Students should experiment!

II. Step-by-Step Guide for the discussion of the AMEISE handout:

There are at least two possibilities: the comparison to objectives of the simulation and a comparison to the average / best / worst of group of simulation runs. A combination of both is suggested. This guide refers to the "evaluation.pdf" file that is produced by the "doit" command of the AMEISE evaluation scripts (Dec. 2004, Andreas Bollin, Susanne Jäger).

1. Reaching the objectives
   → Use self-explanatory colours for reached/missed goals! Be careful with the colours. Table should still stay motivating for the students!

2. Number of developers
   There is a theoretical optimum per project. However, consider
        i. Problems of cost have also to do with number of developers. Theoretically start with only few, hire them during the project and reduce number again at the end
        ii. Often there is NO OPTIMAL SOLUTION
        iii. There is also a LEARNING EFFECT of developers
        A developer needs 2-4 weeks to get really productive
        iv. Be careful with the propagation of knowledge

3. Distribution of effort
    a. Specification ~ 10-15 %
    b. Design ~ 30-40 %
    c. Coding ~25-35 %
    d. Testing ~ 20-30 %
    e. Manuals ~ 10 %

4. Decrease of quality (in respect to AFPs) of documents
    a. Decrease should not be too steep
    b. If quality of code is less than AFP of module specification, then testing not optimal! Solution might be
        i. Improve quality of predecessor documents (not that difficult!)
        ii. Make SW-inspections (VERY EXPENSIVE, but good results)
        Note: THERE IS A TRANSLATION PROBLEM IN SESAM:
        Inspect <…> means: show me information about <…>
        Review code means: CODE inspection
        Review [spec|design|manual] means: review meeting
        iii. Do an acceptance test with customer

5. Management competence of the project manager
    a. It depends on the situation, however:
        i. Too many inspections → lost concentration
        ii. What is "too many"? It depends on your personal philosophy of conducting projects
        iii. In reality it means: it takes time!
    b. But, it is possible to identify "loss of orientation" when the number of inspections INCREASES

6. Developers' productivity
    a. Focus on time when developer is hired but not working
    b. Is there a gap between end of last activity and time when s/he left project? Indicates that student has not planned his/her activities!

7. Consistency of documents
    a. Check, if there is a correct all
    b. Check, if there are several "correct all"-s

8. Influence on consistency of corrective tasks
    a. Look for documents that were produced BEFORE previous documents have been corrected → not a good idea, as errors propagate
    b. Look if there is a correct all

9. Non-productive time
   Rule of the thumb for the QA 200 model: 10.000 € – 13.000 € very good, more than 20.000 € is not really good!

10. Number of errors detected and corrected (in Reviews 2.5 and Tests 2.6)
    a. Write down the NAMES of the developers, reviewers
    b. Look at difference between "Errors found" and "Errors corrected" (a loss of 5-10% is normal, more than 10% indicate problems)
    c. Human beings make mistakes, an CORRECTION introduces NEW Errors
    d. Every time a big difference is identified: LOOK FOR THE REASON They can be identified by LOOKING at diagrams in sections 2.11-2.14

11. Arguing for GANTT Charts (Sections 2.11-2.14)
    a. Creation/Review/Correction should be more or less Waterfall-like
    b. Look at phases that are intersected (more that 50% overlaps lead to problems)
    c. The shaded diagrams represent SUCCESSFULLY completes phases. Are there any actions in the phase that to not belong to a shaded box, it HAS NOT BEEN successful.
    d. Look, if the customer has been involved in reviews

12. Arguing about inspections
    a. Permanent inspection is a good idea
    b. There is no optimal number of inspections. However, an increase of the number of inspections might indicate a LOSS of CONTROL

13. Correctness of documents
    a. The quality should decrease a little bit. However, when quality increases at the level of code, then there has been quite a lot of testing (! Expensive !)
    b. When the quality of code is less than the quality of module specifications, there was a problem with the test

14. Performance
    a. Only displays the number of AFPs and errors in relation to the objectives
    b. It should be approximately 100%. Is it < 100% there are problems, is it >100% then the manager did a great job.

15. Distribution of effort
    a. Effort is displayed in hours
    b. Interesting to look at phases with SAME effort in, but different number of, errors. It indicates whether review and correction did work correctly.
    c. A "0" in a line indicates that no review or correction problem took place.

16. Distribution of effort (Sections 2.16, 2.17)
    Here just look for "0" entries. This means, that (sub)phases have been forgotten

17. Remaining Errors in documents
    a. Good to show that errors propagate in a project
    b. It also demonstrates that new errors are introduced

Final Remarks and Ideas:

In any case it is a good idea to pick DIFFERENT games to show effects. This leads to more interaction and (fruitful) discussion with the students. The steps are then always the same:

- Pick interesting diagram
- Identify the reasons (direct/indirect effects) leading to the result
- Discuss it in the feedback session

As a variation it is possible to ask students to do a self-evaluation of the results of their own (or others) second simulation run. You can also ask them to search for literature to underpin their arguments.

The 2nd game usually is used by students to optimize their results. However, when a 3rd simulation takes place, let them experiment. Provide the protocol of their 2nd run and ask them to change some decisions (in order to identify/experience the difference).