Design of CQA Systems for Flexible and Scalable Deployment and Evaluation

Ivan Srba, Maria Bielikova
Faculty of Informatics and Information Technologies
Slovak University of Technology in Bratislava
Community Question Answering (CQA)

• Communities of millions of users share their knowledge
  • by providing answers
  • on questions asked by the rest of the community

• CQA systems have been adapted into additional contexts and environments
  • Educational domain
  • Crowd-based customer services, integrated development environments (IDEs)

• Initial research how specifics of these new environments affect
  • Essential features (e.g. core question answering functions)
  • Collaboration support (e.g. question recommendation)
Open Problems

1. **Low adaptability of essential features to various settings**
   - Possibility to be deployed in several different instances at the same time (e.g. in several educational or enterprise organizations)
   - How to make design of essential features flexible to handle various settings?

2. **Ineffective integration and evaluation of collaboration support methods**
   - Possibility to perform live experiments (very rare in standard CQA systems)
   - How to achieve loosely coupled integration of collaboration support methods with CQA systems?
   - How to make combination of offline and online experiments as effective as possible?
**Case Study on Our CQA System Askalot**

- A novel concept of an organization-wide educational CQA system

- 1150 users
- 430 questions
- 560 answers
- 410 comments

Ruby on Rails
Open source
Case Study on Our CQA System Askalot

• The version of Askalot was proposed specifically for our university
  • ... and thus it lacked sufficient flexibility and scalability

• We started a cooperation with:
  1. Harvard University in order to transform Askalot into a plugin to MOOC system edX
  2. University of Lugano in order to deploy at additional universities as a part of cooperation project in the SCOPES program
* Designing Essential Features for Various Settings

- Modular system architecture
- Adaptable self-managed content organization
- Flexible user management integration
- Ubiquitous activity awareness and notifications
Modular System Architecture

• Necessity to develop two main configurations of our system
  • Askalot @university
  • Askalot @mooc

• Solution based on one application and three components (RoR engines)
  • Shared
    • core features that are common for both configurations
  • University and MOOC
    • inherit all features from the core component and add specialized features
Adaptable Self-managed Content Organization

• Two-level topic structure which supports easy and flexible deployment
  • Category level
    • Reflects the formal structure of a university or a MOOC course
  • Tag level
    • Selected by asker to describe particular question topics

• Categories should
  • be hierarchical
  • reflect repeating sessions (i.e. academic years or course sessions)

• Solution based on nested set pattern
  • Hierarchical tree where each node has
    • domain-specific ID - to identify the same categories across all academic years or course sessions
    • shareable flag – whether questions from the previous sessions should be displayed also in the current session
    • askable flag – whether students can ask questions in this category
• A: Global view loaded from shared component
• B: Unit view loaded from MOOC component
Designing Universal Experimental Infrastructure

- Modular approach where all methods are loosely coupled from other methods or system itself
- Possibility to combine training/evaluation of methods on offline datasets with live experiments
Three main parts of experimental infrastructure

- Stack Exchange Dataset (XML format)
- Askalot (live system)
- Stack Exchange Dataset (database)

**Resources**

- Stack Exchange XML-DB Convertor
- Event Simulation Job
- Event Dispatcher
- User/Content Profilers
- Method Feeders & Evaluators
- User/Question/Answer Profiles

**Dataset Conversion**

**Event Dispatching**

**Listeners and Profiles**
Part 1: Dataset Conversion

- Utilities to convert
  - any datasets from CQA systems
  - to a dedicated experimental database with the same database schema as Askalot system

- Existing convertors
  - Stack Exchange datasets
  - edX datasets (in progress)
**Part 2: Event Dispatching**

- **Event is represented by**
  - a resource
    - question, answer, comment, view, vote
  - an action type
    - create, update, delete
  - an initiator
    - who performed this action

- **Event sources**
  - live system in online experiments
  - datasets in offline experiments
    - either from Askalot itself
    - or from other CQA systems
Part 2: Event Dispatching

- Event simulation job
  - selects from the database all resources
  - converts them to a list of events
  - sorts events by time when they originally happened
  - sets the current time in the experimental environment to this event time
  - dispatches the event

- Exact reproduction of events
  - as they would be created by the live system
Part 3: Listeners and Profiles

- Listeners
  - Profilers - model users/content
    - user expertise, question difficulty
  - Method feeders - trigger and evaluate various research methods
    - recommendation of new questions to potential answerers

- User/question/answer profiles
  - Universal data structures to store results of profilers
    - attribute
    - value
    - probability
    - source
Conclusion

• Drawing upon redesign of CQA system Askalot, we proposed several design recommendations
  • how concepts of CQA systems can be adapted to an educational context and organizational environment
  • with achieving high flexibility and scalability

• As the result, Askalot can be
  • deployed in three different environments (and many more if necessary)
  • characterized also as an open platform based on the universal experimental infrastructure

🏠 askalot.fiit.stuba.sk/demo (Try it!)

GitHub: github.com/AskalotCQA/askalot
Email: askalot@fiit.stuba.sk