Presentation of
“Hipikat: A Project Memory for Software Development”

Lucas Panjer

October 5, 2006

Summary

- How to integrate new developers into a project
- Studies show that non-collocated teams do not collaborate as effectively as collocated
- Yet distributed teams have many means of collaborating.
  - Source repository, CVS, SVN, etc
  - Mailing lists, email, chat logs
  - Bugzilla, trac, etc
  - Wiki, project documentation
- Missing project memory?

Proposed solution

- Recommendation system
- Live data mining
- Implicit link generation
- Client-server architecture
  - Shared DB
  - Standardized communication interfaces (XML-RPC, SOAP,…)
- Integrated with software engineering repositories

Server modules

- Artifact Database
- Update methods per repository
- Heurisitic Link identification
  - Modules monitor
    - Check-in logs
    - Activity correlation (check-in, bug closure)
    - Text similarity (calculated for indexing)
    - CVS check-in linking
    - Thread matching

Client

- Eclipse plug-in
- Text search
- Contextual search
- Estimate of relevance
- Artifact inspection
- Rating / Confidence

Quality Study

- Selected bugs with severity = low (n=215)
- Randomly selected 20 of those
  - Discarded obvious non-bugs
- Precision: Fraction of recommended files that contribute to the solution
- Recall: Fraction of files in solution that are recommended
- Precision: 0.11
- Recall: 0.65

User Testing

- Evaluating newcomer task completion
  - Two real scenarios selected, with optimal outcomes. Easy task and Difficult task
  - Participants: Newcomers, Experts
- Hipikat accessed more during
- Results: Newcomers perform about as well as experts

Contributions

- Novel platform for recommendations in SE repositories
- Extensible
- Broad application to OSS projects
- Learning tool
- Design recovery
- Common error avoidance

Surprises

- Relatively high recall
- Newcomers could perform about as well as experts
- Only used for planning

Applications

- Collaborative filtering would likely drastically improve this tool
- Extraction of knowledge into documentation
  - Cluster, then ask for a description
- Valid for non-OSS projects?

Positive

- Good start for recommendations
- Exposes reasoning and confidence
- Extensible architecture
  - Plug-in heuristics, repository clients

Negative

- Is project memory actually just related artifacts?
- Dependent on unenforceable conventions
- Selection of test cases (severity = low)
- Number of test cases
- Huge tradeoff between precision and recall