Web browsing support for cross-community activities

Tomohiro Oda
Agenda

• cross-community activity
• cross-community activity and DynC
• difficulties in supporting cross-community activities
• cSuite: web browsing support tool for cross-community activities
• cSuite for DynC
Cross-community activity

**Definition:**

- Activity either
  - needs support of multiple communities, or
  - contributes to multiple communities.

**Examples:**

- standard graph format
- developing OpenGL interface in Smalltalk for CAD system of a ship constructor.
Communities, individuals, activities, and interests
Comparison with DynC

**Similarities**
- Focused on each individual's tasks or activities
- not for community, but for individuals

**Differences**
- supportive community v.s. supportive person
- assuming pre-existing communities v.s. forming a new short-term community
Difficulties in supporting cross-community activities

• **A task needs knowledge of multiple communities.**
  - None of each community covers the whole task.
  - It is hard to identify/describe the task from each community's viewpoint.

• **It is difficult to recommend collaborators/related artifacts**
  - different motivations, interests, and goals on a same topic
Example difficulties: web browsing

- **difficult to identify tasks**
  - Browsing a community's website does not mean the user is working on a task covered by the community. 
    e.g. A CAD programmer is reading the C-99 specification. Does the C language community cover CAD programming?

- **difficult to recommend collaborators/related artifacts**
  - Browsing the same document does not mean sharing the same interests and goals. 
    e.g. Two programmers are reading HOW-TO of Linux-2.6 device driver. One is a FreeBSD kernel hacker, and another is an ethernet board manufacturer.
cSuite: cross-community support using HTTP proxy

- Each community provides "glossary" as community's knowledge.

- A user specifies a list of glossary servers that the user is interested in.

- cSuite provides additional information to HTML documents.

The development of cSuite is sponsored by IPA, Japan.
Basic ideas of cSuite

• One possible way to identify user's task and to find supportive persons/related documents:
  − Words are very important clues of user's tasks.
  − Many communities provide their glossaries as
    • FAQs
    • Tutorials
  − Natural Language Processing techniques like
    • Text classification
    • Word disambiguation
Architecture of cSuite

- cScope
  - HTTP proxy
- cSorter
  - info recommender
- cIris
  - message filter
- glossary
- bookmark folders
- user model (Naive Bayes)
- Information (URL history)
- localhost
- WebBrowser
cScope: HTTP proxy

- cScope is a private HTTP proxy server which works on localhost.
- cScope wiretaps all "GET" requests and returned HTML documents.
- cScope inserts icons to each occurrence of keywords.
- Each icon represents a community.
Context delivery
cSorter: datamining user's interests

- A user provides "categories", which represents user's interests.
- The user also gives bookmarks in each category, which are sample documents of each interest.
- cSorter recommends documents for each category using Naive Bayes (from URL history).
Interests are dynamic

- The system should catch up updates of user's interests.
  - A user may get interested in a new topic.
  - A user may expand the range of a topic.
  - A user may retract a topic of interest.
  - A user may have different interests on a same document.
  - and so on...
cIris: Information filter at end points

- Many communities provide tons of information via mailing lists.
- Many participates have only partial interests in the community's topics.
- cIris filters documents using the stochastic model developed by cSorter.
- cIris uses distribution of keywords as a user model.
  (similar to distribution of functionality)
Suppose that you are sending a message to a mailing list...

- A sender don't know receivers' interests.
  - You may hesitate to broadcast the message which many receiver can respond to.
  - Or, you may bother people by broadcasting the message which no reciever really care.
- Using cIris, senders don't have to worry about receivers' interests.
Difficulties revisited

• Identifying task
  – cSorter can classify recent N documents to identify the topic of the current task.
  – cScope can help users to identify potential topic of the current task.

• Recommending artifacts/people
  – cSorter
  – cIris: see the next slide.
cSuite for DynC

• Possible ways to extend cSuite for “dynamic community"
  – Use cIris to screen persons
    • public cIris: Send remote query to cIris of your friends.
      – privacy issue ... cIris has a lot of private information!
  • P2P cIris: Flood the message into a P2P-like network and filter at each node using cIris.
Conclusions

• Cross-community activities need support over multiple communities.

• cSuite is a support tool for cross-community activities focused on individuals:
  – Context delivery suggests potential support of / potential contribution to a community.
  – Document categorization catches up changes of interests.
  – Information filtering at receiver's end.

• Possible extension for DynC