Cognitive and Social Support for Learning Java API

Yunwen Ye

April 24, 2006

University of Colorado at Boulder

Guest Lecture at Design, Learning and Collaboration

I have a dream…

In 1983, 85% of the code has been repeated by someone else in the world. (Capers Jones, 1984)

Every time when I am trying to re-invent the wheel, an agent brings me the existing wheel that I can use immediately

I have yet another dream…

All the information I need is always at my desktop, virtual or real; and the needed information only

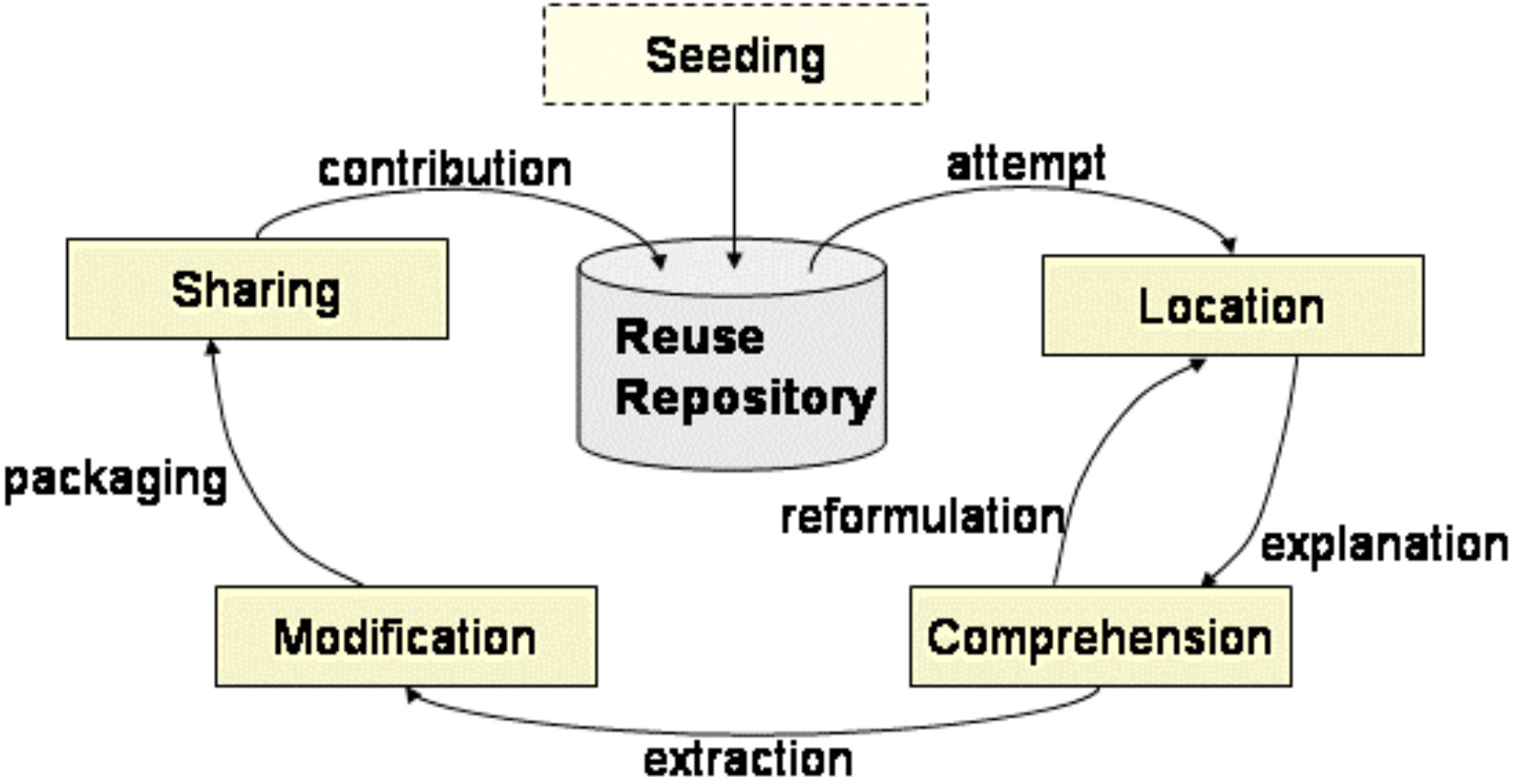
Software reuse

- Definition
  - Creating new software systems with existing artifacts
- Reusable artifacts
  - Code artifacts
    - macros, functions, ***methods, classes***, subsystems, systems (Open Source Software)
  - Non-code artifacts
    - analyses, designs, test plans and cases, domain models
  - Knowledge
    - program idioms, program plans, design patterns, software architecture styles, domain knowledge
- Reuse repository systems
  - Supporting reuse activities

Why reuse?

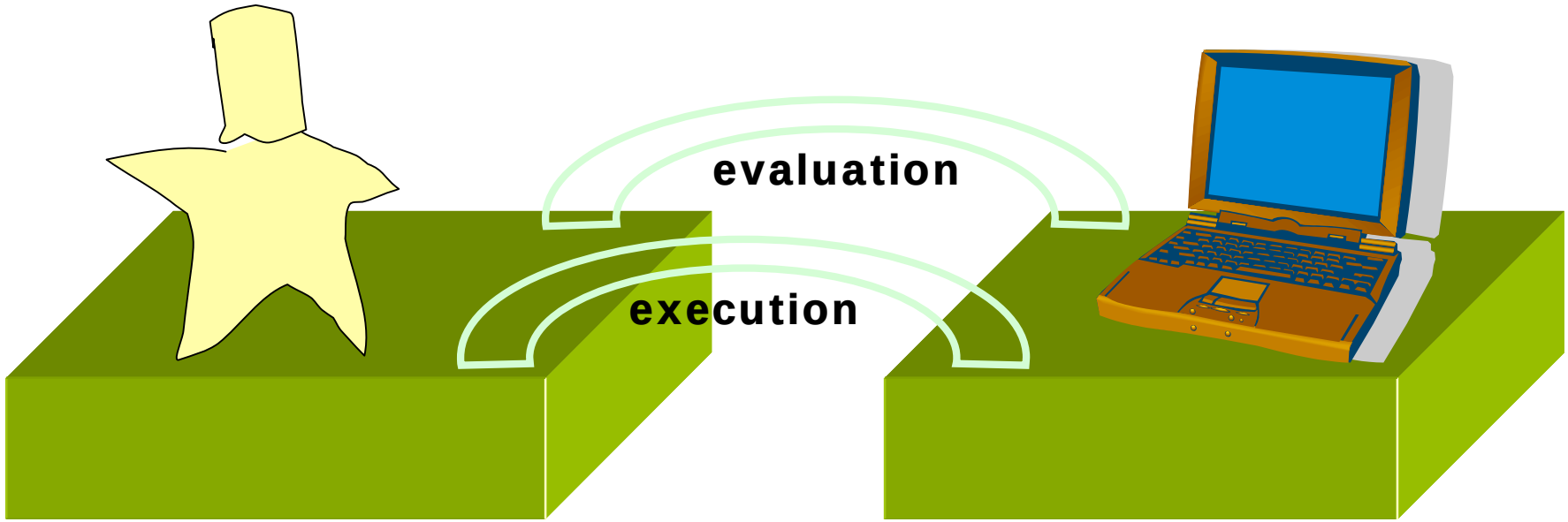
- Increased productivity
  - Reduced development time
  - Reduced cognitive load
  - Reduced testing time
- Increased quality
  - Fewer bugs
- Enhanced evolvability and maintainability

Reuse process (sLCMS)



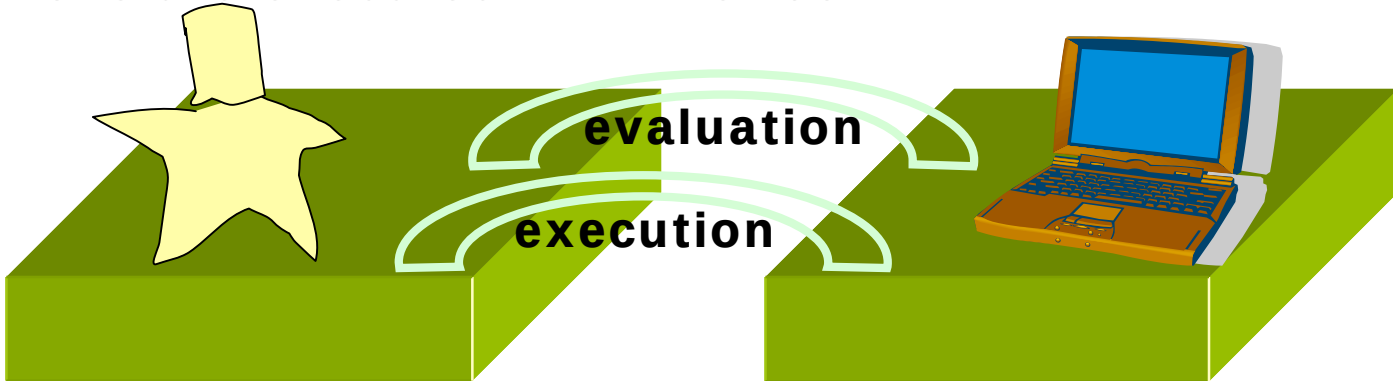
Understanding the cognitive issues in reuse

- Cognitive engineering:
  - Apply what is known from cognitive science to the design and construction of tools that assists cognitive activities of human beings
- Bridge two gulfs between users and tools



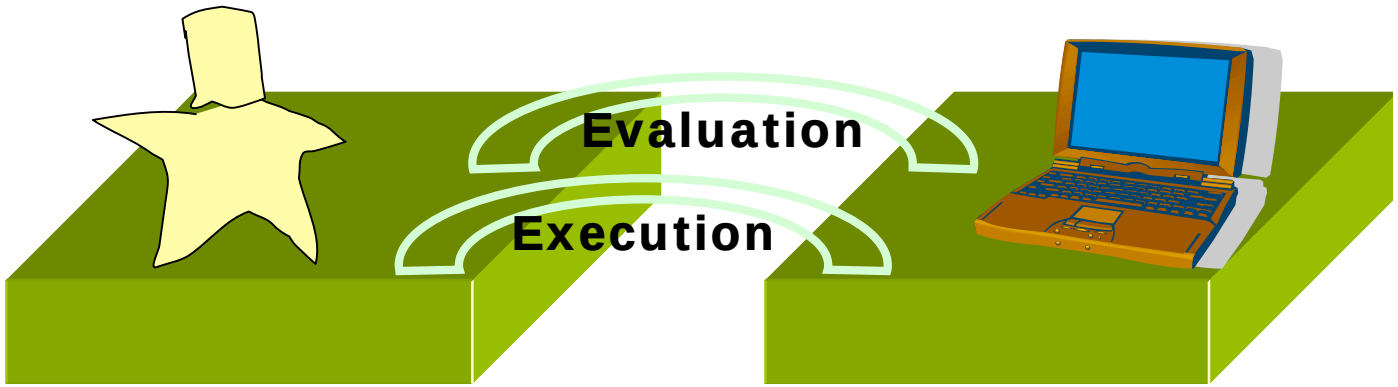
Execution gulf

- Bridging the gap from the goal to the tool
  - Intention Formation
    - Users decide to do something with an internal specification of the task created from their goal.
  - Action Specification
    - Users externalize the internal specification into a sequence of specified actions.
  - Action Execution
    - The actions are executed with the tool.

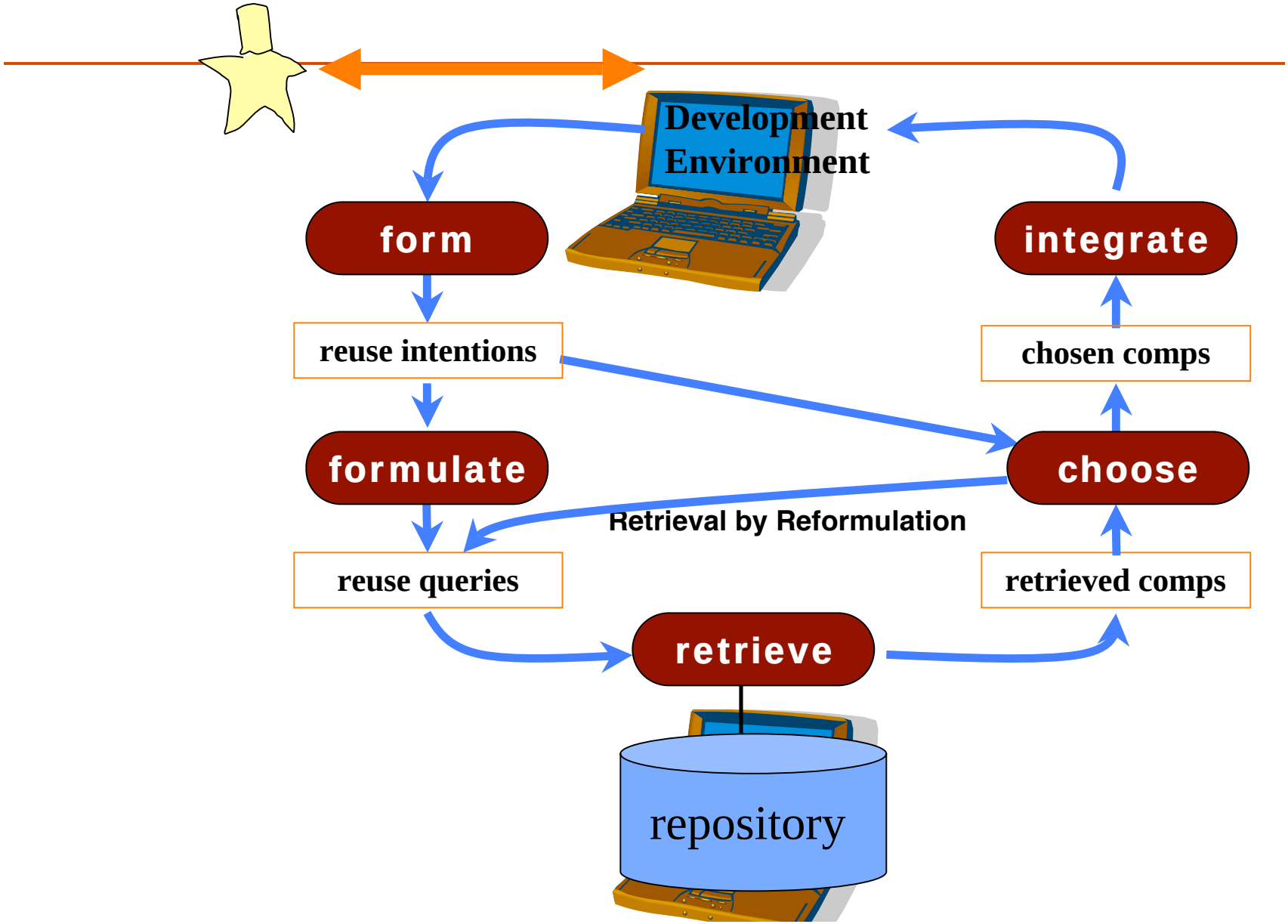


Evaluation gulf

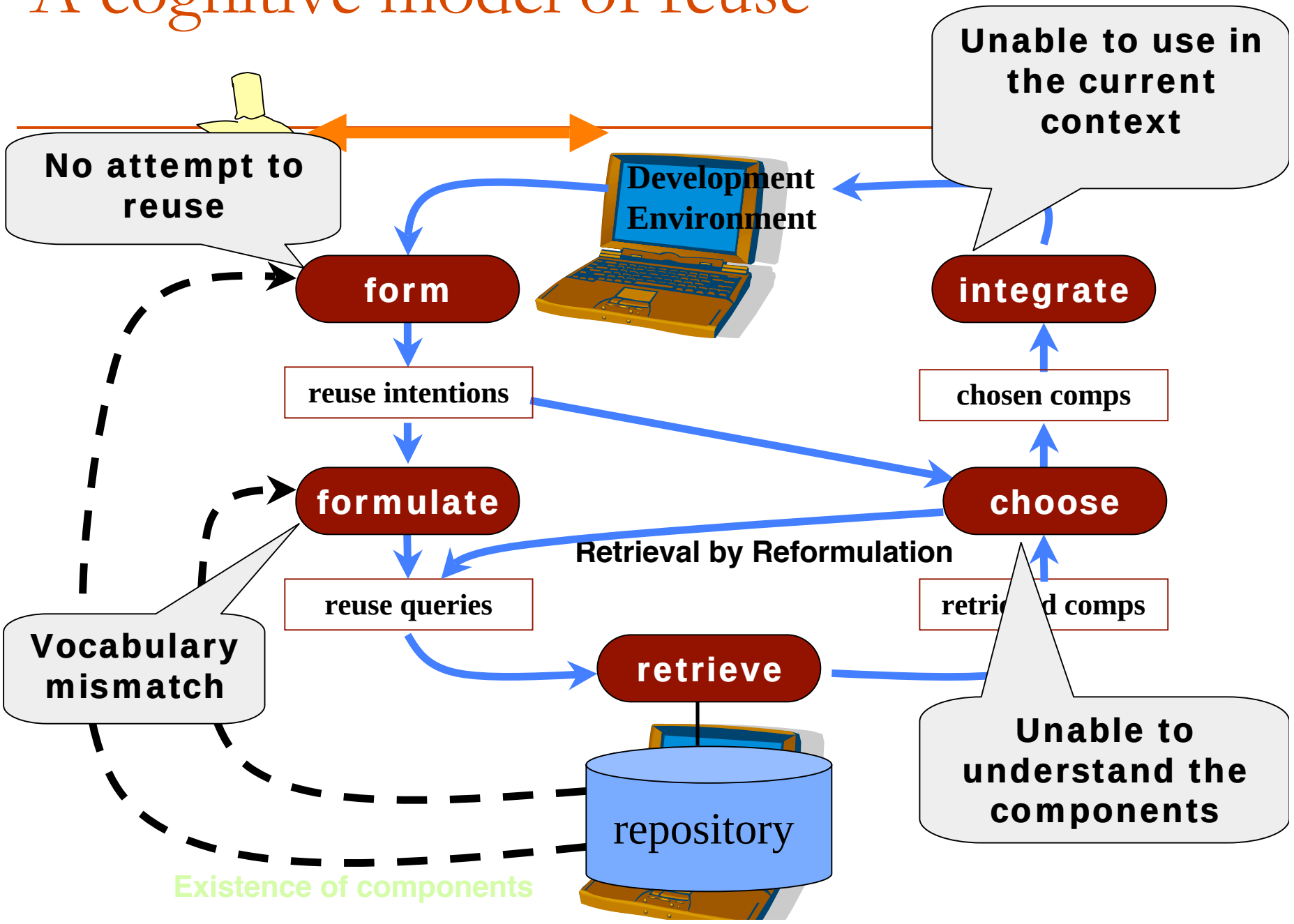
- Bridging the gap from tool output to goal
  - System Perception
    - Users perceive the output of the tool.
  - Interpretation
    - Users interpret the perceived output.
  - Evaluation
    - Users compare the interpretation with the original goal.



A cognitive model of reuse



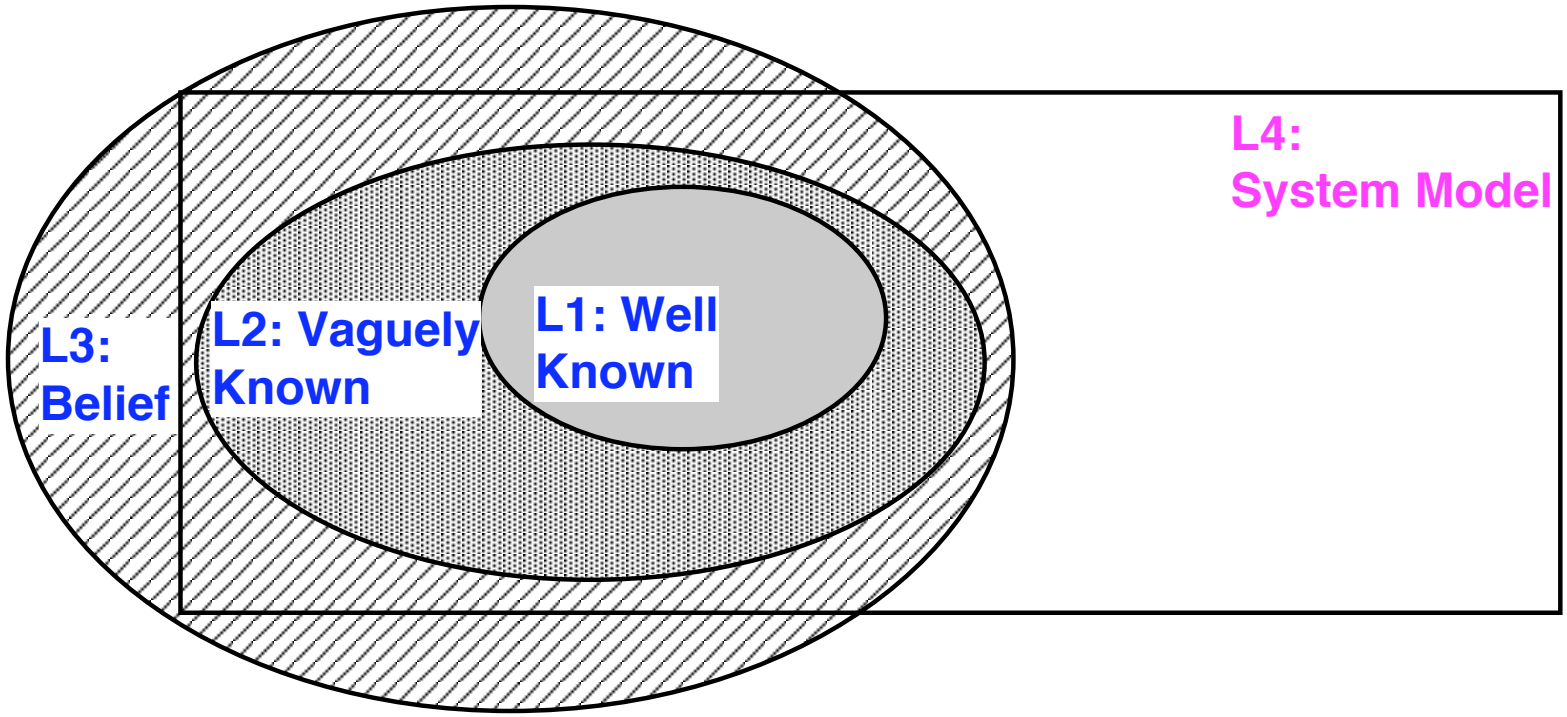
A cognitive model of reuse



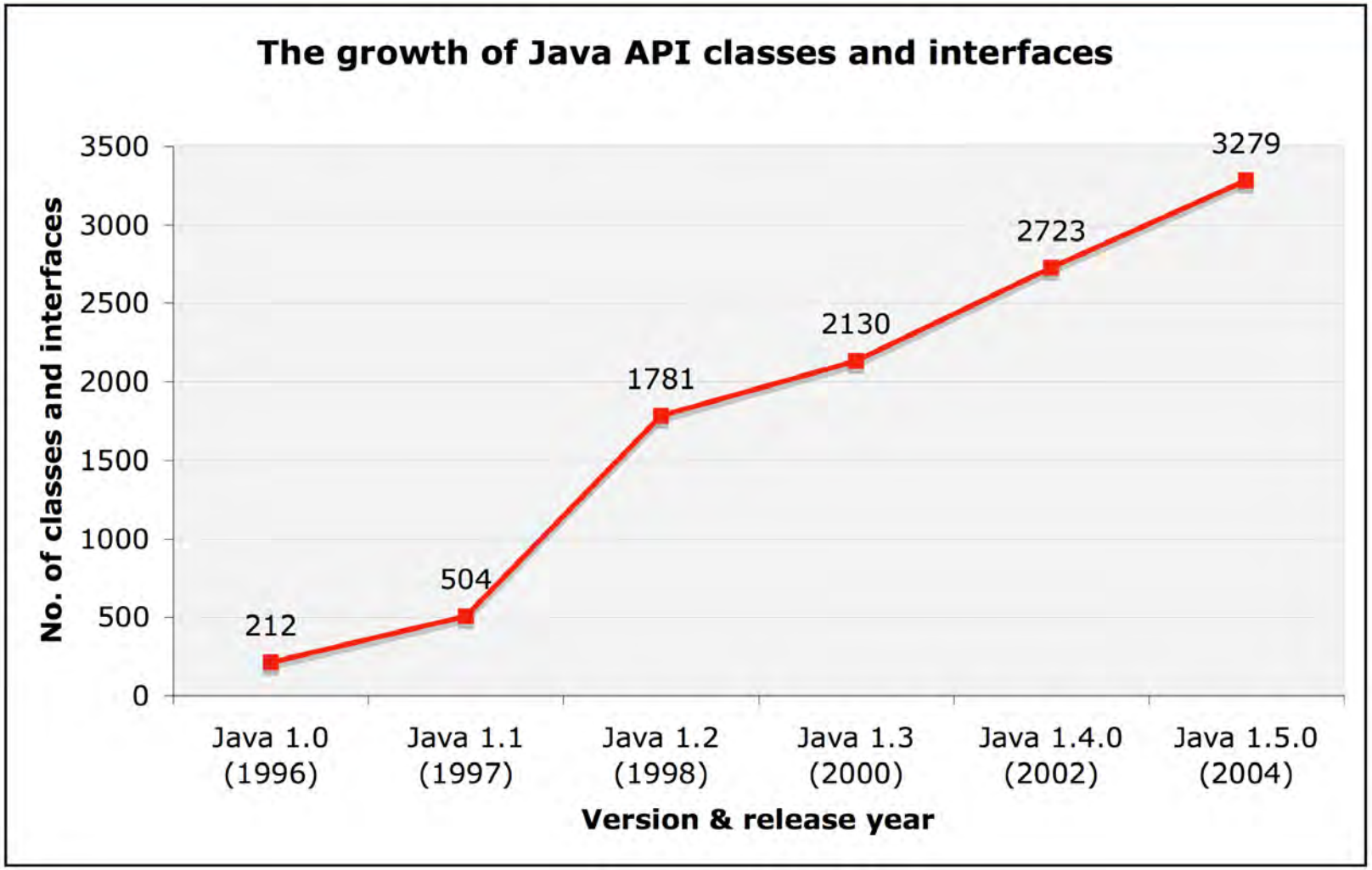
Research problems

- No attempt to reuse (*Location*)
  - Information islands
    - Not aware of the existence of reusable components
  - Perceived low reuse utility (benefits/cost)
    - High cost of locating components
- Unable to locate the component (*Location*)
  - Situation model vs. system model
- Unable to use the component (*Comprehension*)

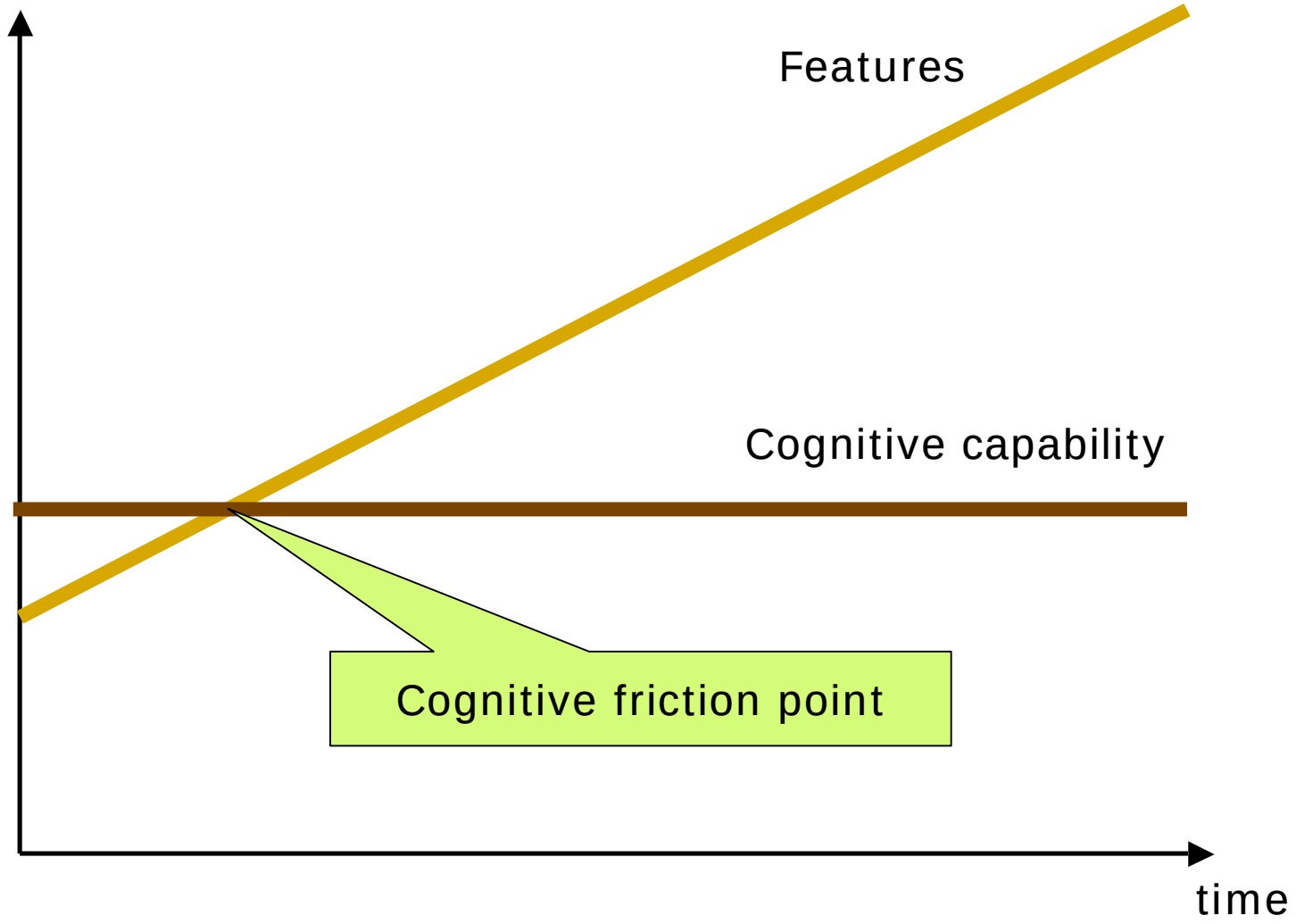
User's knowledge about a reuse repository



Growth of Java class library



Functionality of development tools

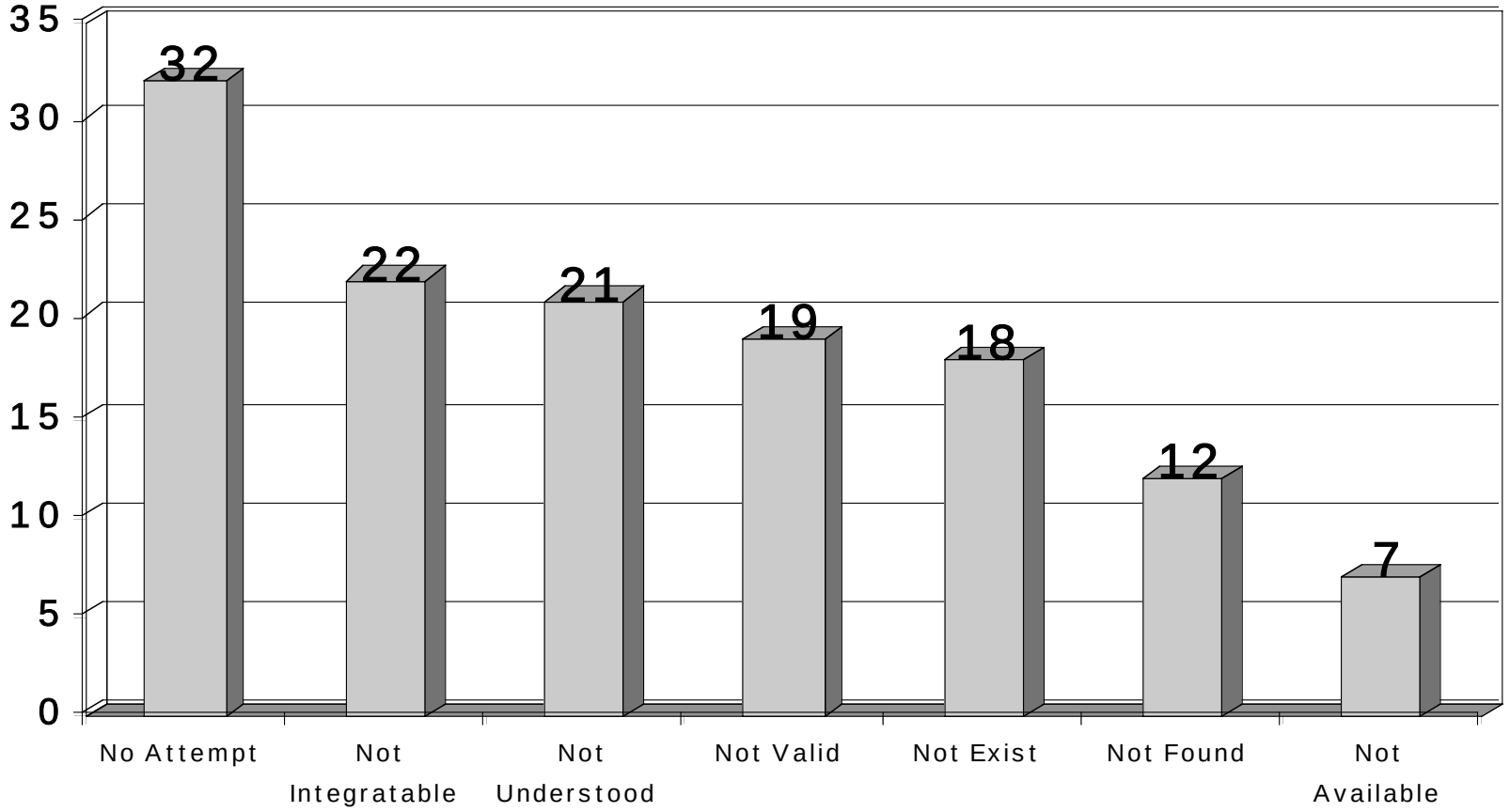


Libraries used in STeP_IN

Library Name	Class	Method
activation.jar	38	261
bcel-5.1.jar	373	3093
commons-collections-3.1.jar	446	4021
commons-dbcp-1.2.1.jar	44	935
commons-pool-1.2.jar	25	277
jun547.jar	2640	18412
mail.jar	240	1966
postgresql.jar	82	1216
resolver.jar	29	298
StPL75.jar	175	1384
xercesImpl.jar	784	7463
xml-ParserAPIs.jar	207	1748
Total	**5083**	**41074**

No attempt to reuse

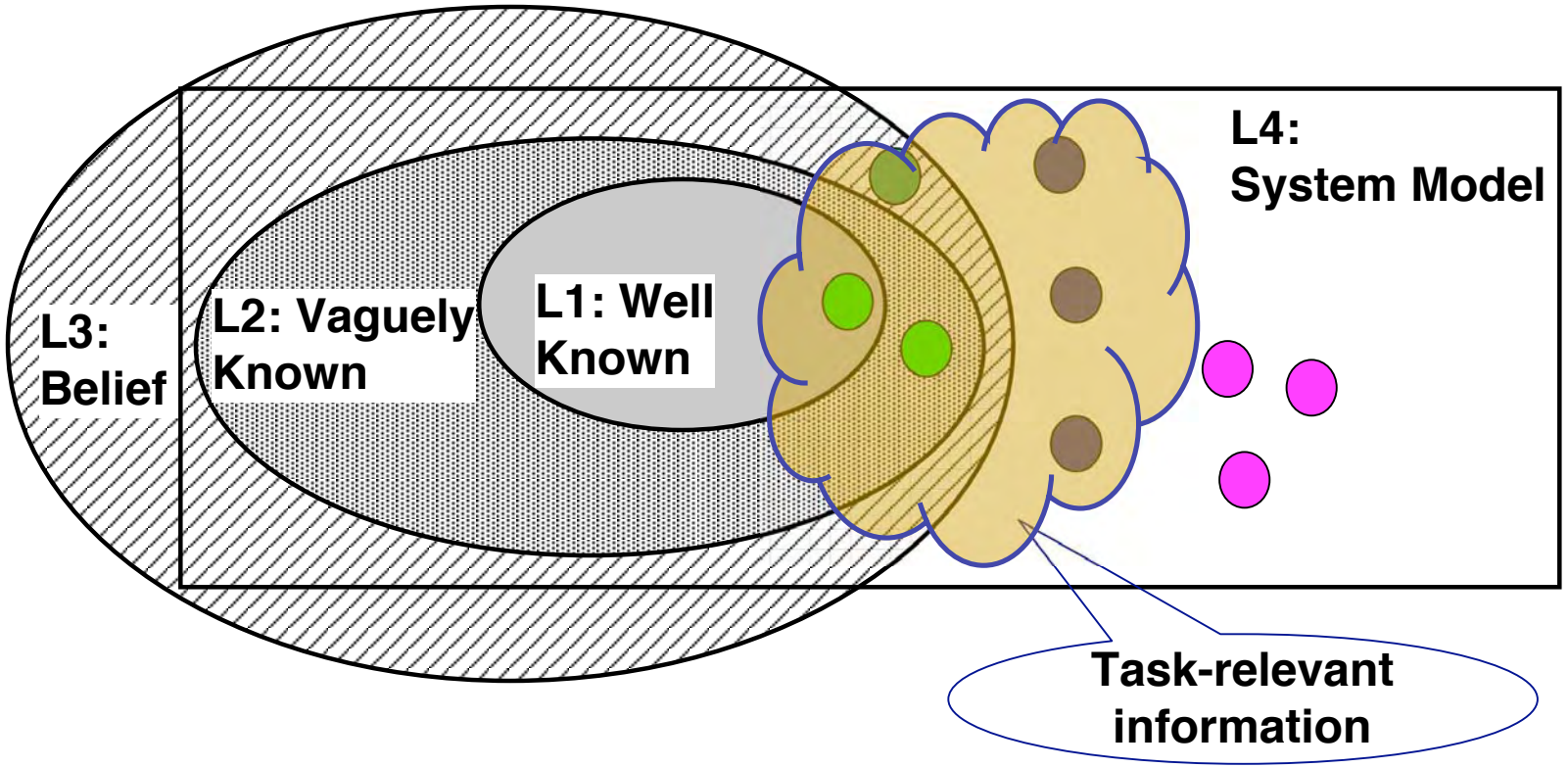
- No attempt to reuse is the most significant barrier to reuse (Frakes & Fox, 1996)



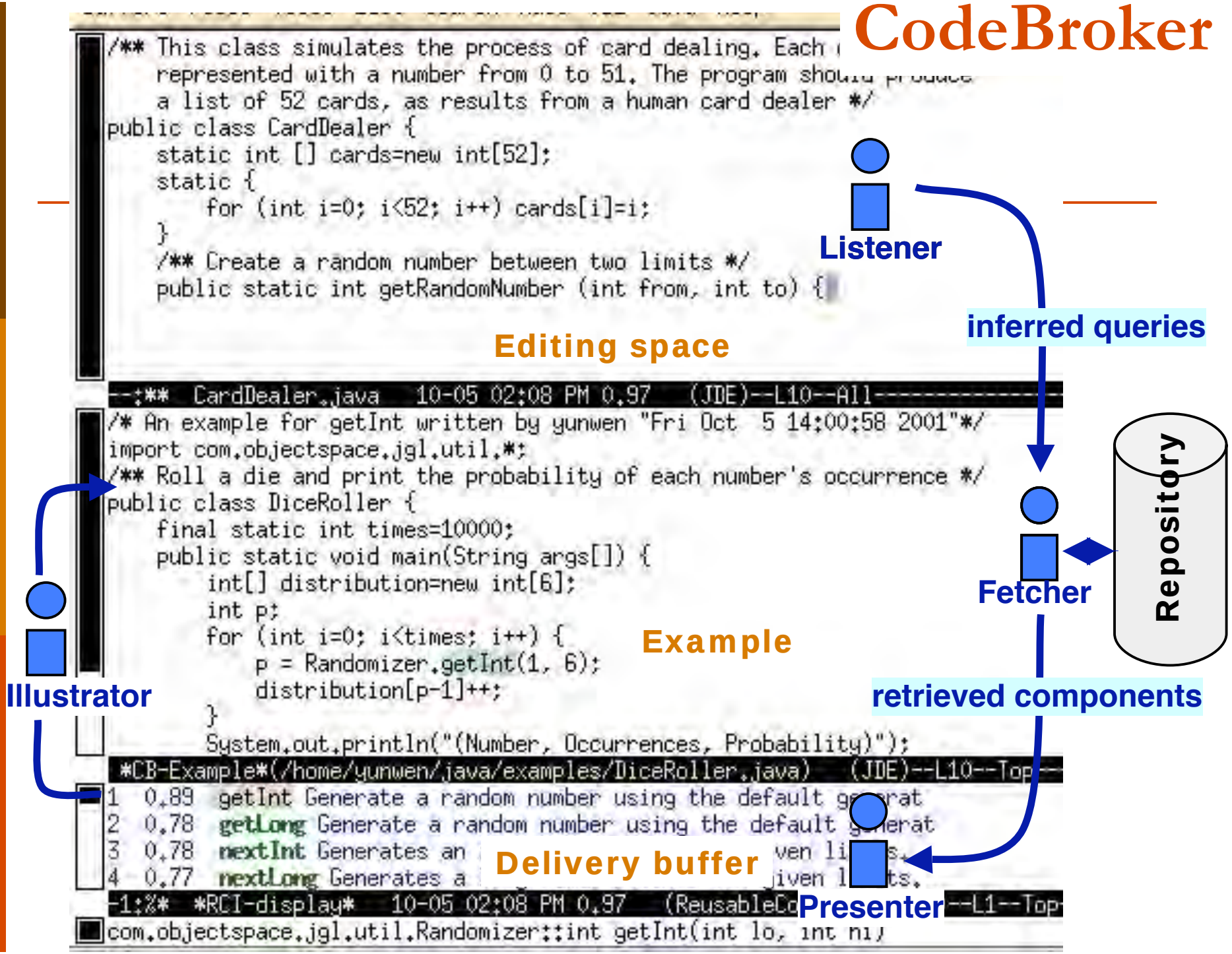
Proposed solution

- Active component repository systems
  - Overcoming the limits of browsing and searching
  - Supporting information delivery
- Benefits
  - Reusing unknown components
  - Reduced locating cost
  - Seamless integration with programming environment

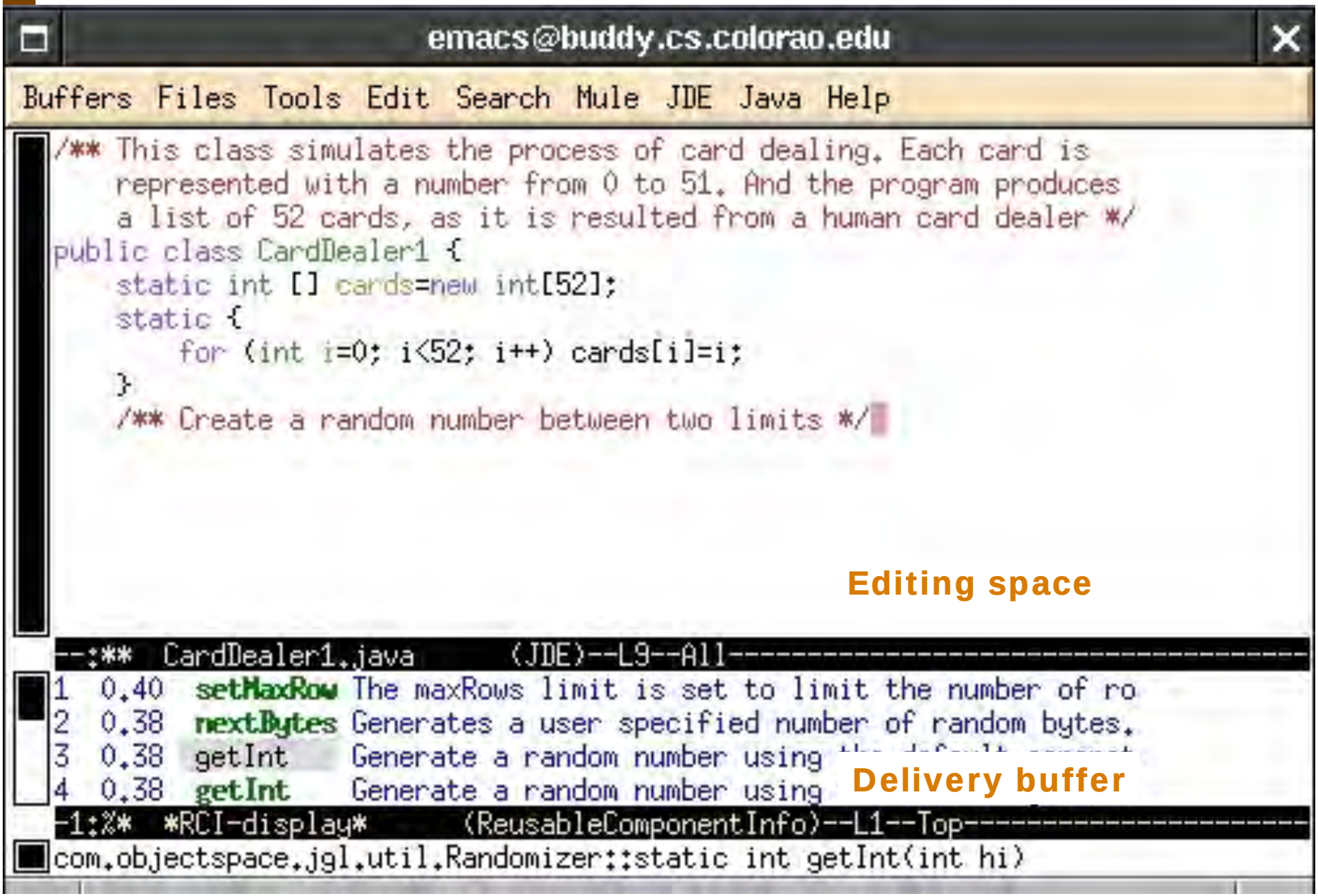
Challenges in active reuse repository systems



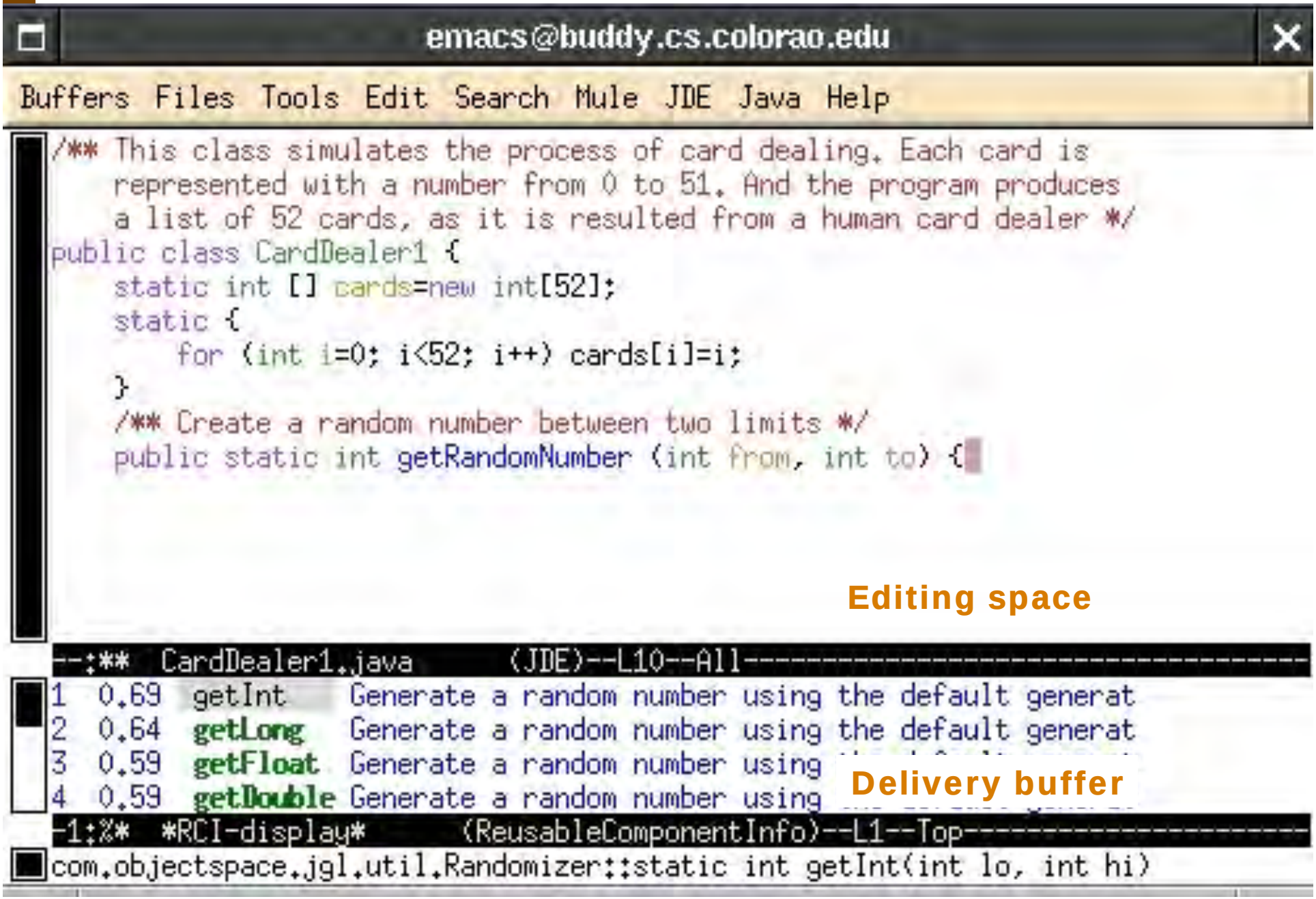
CodeBroker



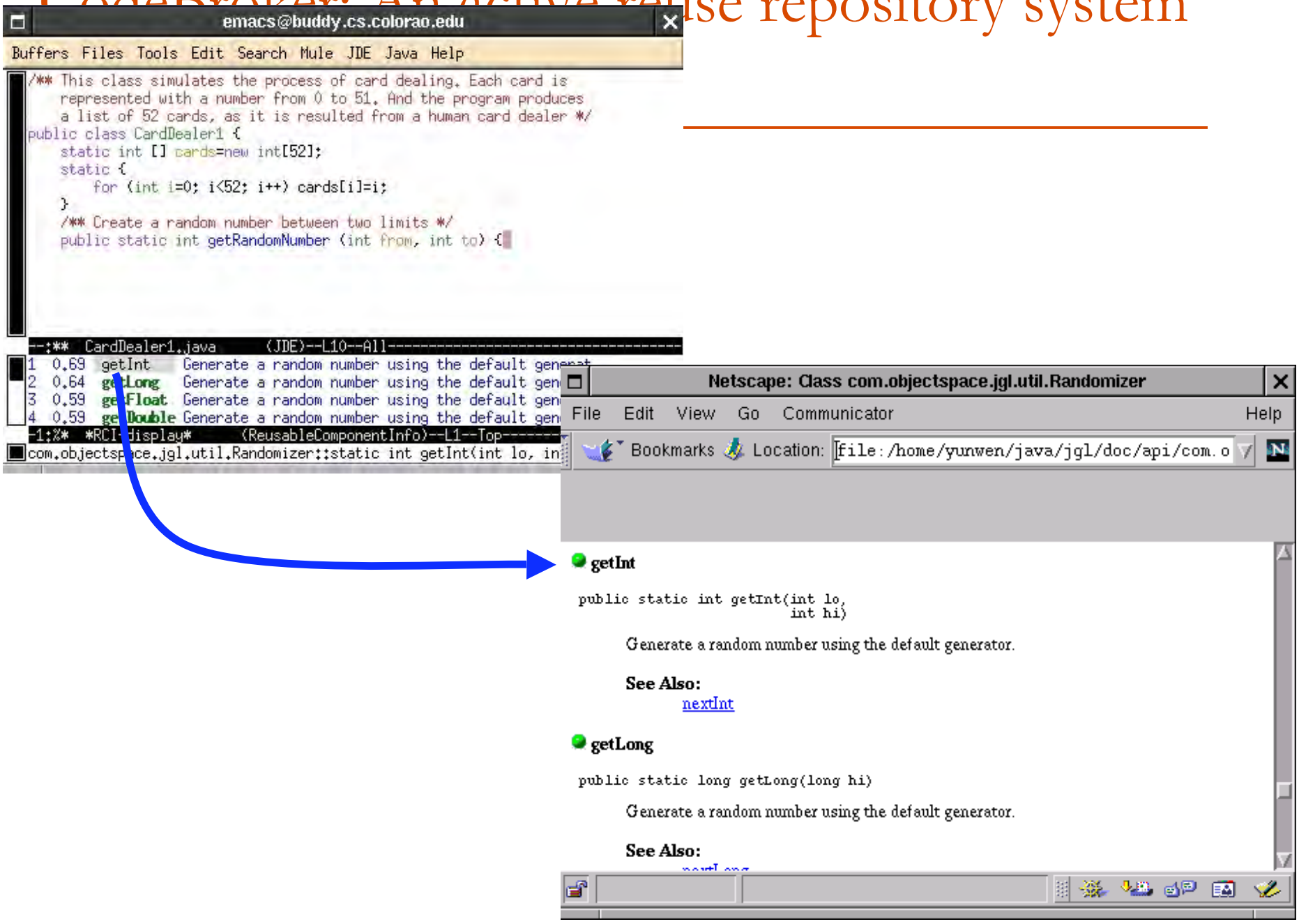
CodeBroker: An active reuse repository system



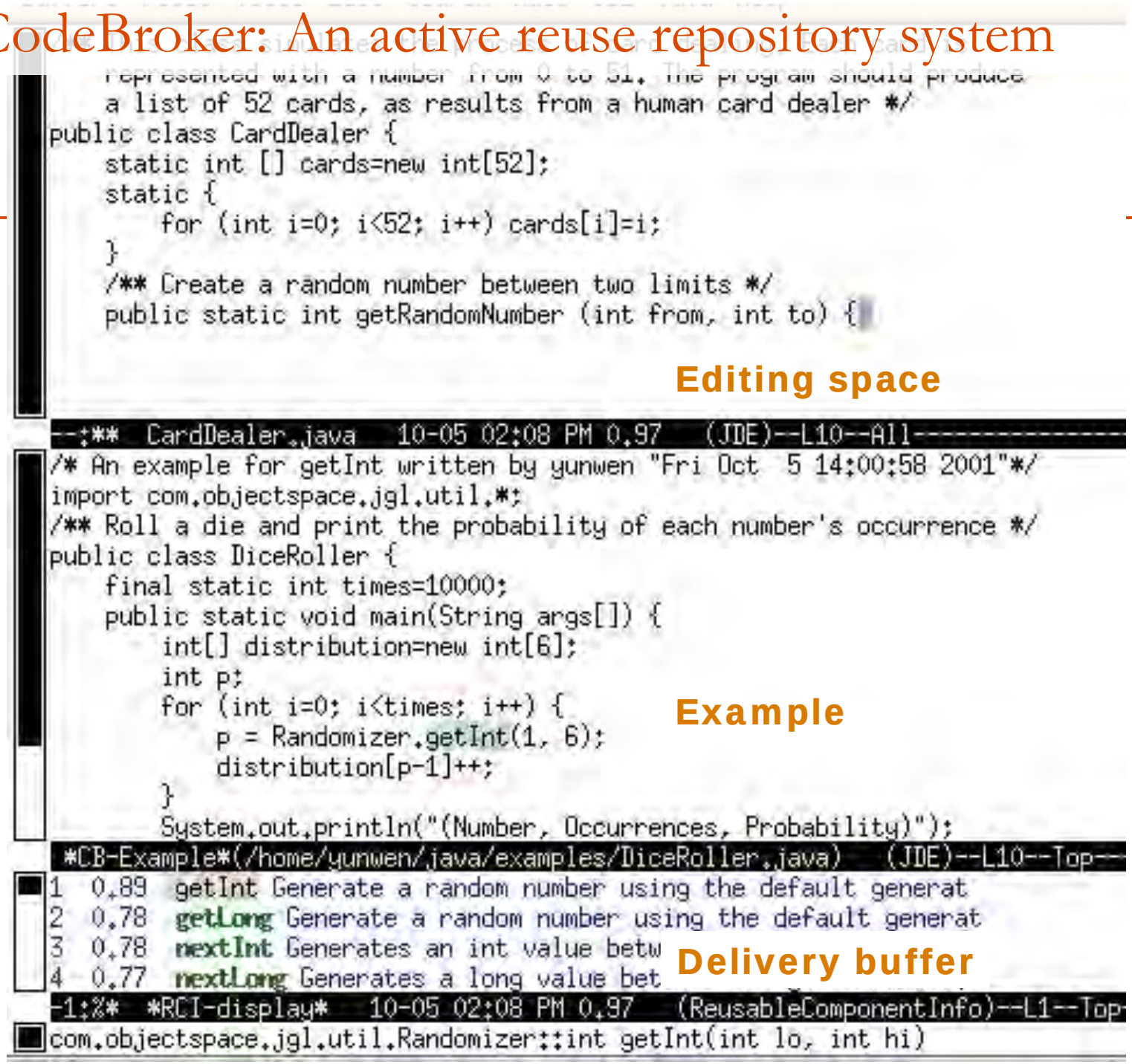
CodeBroker: An active reuse repository system



CodeBroker: An active reuse repository system

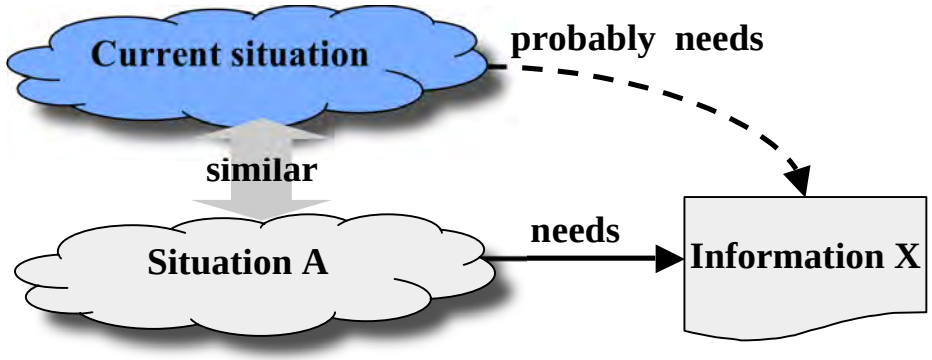


CodeBroker: An active reuse repository system

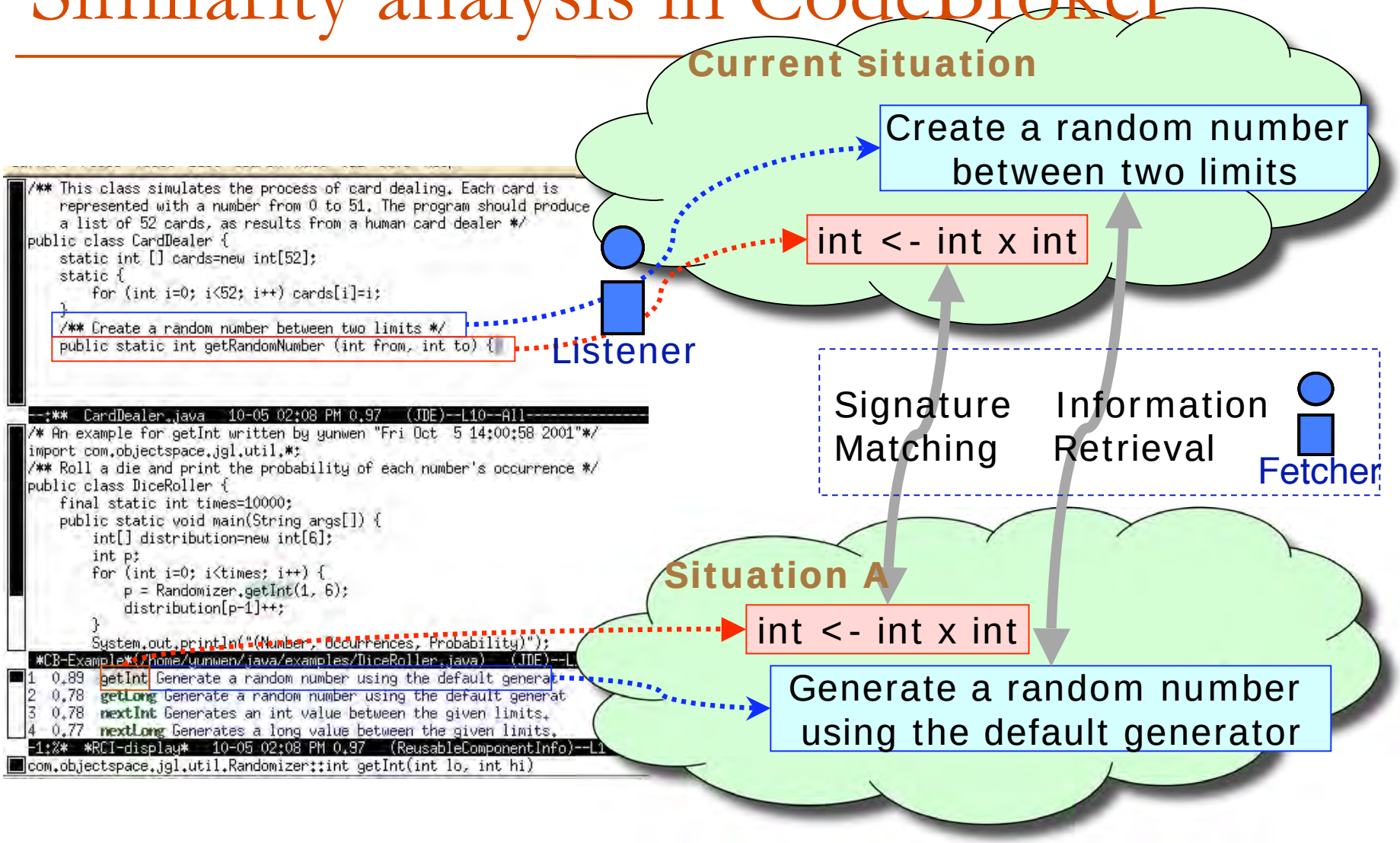


Inferring the task

- Plan recognition
  - Actions → Inferred goal → Suggested actions or information
- Similarity analysis



Similarity analysis in CodeBroker



The rationale

- Three aspects of a program
  - Concept
    - The functionality of the program
    - Semantic information
    - Revealed in *comments*, identifiers, …
  - Constraint
    - Execution environment
    - Syntactic information
    - Revealed in *signatures*, protocols, …
  - Code
    - The implementation
- The assumption
  - Similar concept + compatible signature → reusable code

Basic information retrieval (IR) techniques

- Information retrieval: Finding similar documents based on the commonality of terms
  - Documents and queries are represented by term vectors

    $D_j = (f_{1,j}, f_{2,j}, \ldots, f_{N,j})$
  - Similarity is the distance between two vectors

$$Similarity(Q,D) = \sum_{i=1}^{n} Q[i] \times D[i] \Big/ \sqrt{\sum_{i=1}^{n} Q[i]^2 \times \sum_{i=1}^{n} D[i]^2}$$

Term space: (factor information help human operation retrieval system)

	Contents	Vector	Similarity
Q	human factors in information retrieval system	(1 1 0 1 0 1 1)	
D1	factor factor factor human human retrieval system	(3 0 0 2 0 1 1)	$7/75^{0.5}=0.80$
D2	information operation retrieval retrieval	(0 1 0 0 1 2 0)	0.55
D3	factor help help retrieval	(1 0 2 0 0 1 0)	0.37

LSA: Improved IR

- Latent semantic analysis
  - Addressing the vocabulary mismatch problem (people use different names to refer to the same concept)
  - Creating a semantic space with a large amount of documents

$$\begin{pmatrix} w_{1,1} & w_{1,2} & \dots & w_{1,M} \\ w_{2,1} & w_{2,2} & \dots & w_{2,M} \\ \dots & \dots & \dots & \dots \\ w_{N,1} & w_{N,2} & \dots & w_{N,M} \end{pmatrix} = \begin{pmatrix} t_{1,1}^{(0)} & t_{1,2}^{(0)} & \dots & t_{1,r}^{(0)} \\ t_{2,1}^{(0)} & t_{2,2}^{(0)} & \dots & t_{2,r}^{(0)} \\ \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots \\ t_{N,1}^{(0)} & t_{N,2}^{(0)} & \dots & t_{N,r}^{(0)} \end{pmatrix} \times \begin{pmatrix} s_{1,1} & 0 & \dots & 0 \\ 0 & s_{2,2} & \dots & 0 \\ \dots & \dots & \dots & \dots \\ 0 & 0 & \dots & s_{r,r} \end{pmatrix} \times \begin{pmatrix} d_{1,1}^{(0)} & d_{1,2}^{(0)} & \dots & d_{1,M}^{(0)} \\ d_{2,1}^{(0)} & d_{2,2}^{(0)} & \dots & d_{2,M}^{(0)} \\ \dots & \dots & \dots & \dots \\ d_{r,1}^{(0)} & d_{r,2}^{(0)} & \dots & d_{r,M}^{(0)} \end{pmatrix}$$

Reducing the singular vectors

$$\hat{X} = \begin{pmatrix} t_{1,1} & t_{1,2} & \dots & t_{1,k} \\ t_{2,1} & t_{2,2} & \dots & t_{2,k} \\ \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots \\ t_{N,1} & t_{N,2} & \dots & t_{N,k} \end{pmatrix} \times \begin{pmatrix} s_{1,1} & 0 & \dots & 0 \\ 0 & s_{2,2} & \dots & 0 \\ \dots & \dots & \dots & \dots \\ 0 & 0 & \dots & s_{k,k} \end{pmatrix} \times \begin{pmatrix} d_{1,1} & d_{1,2} & \dots & d_{1,M} \\ d_{2,1} & d_{2,2} & \dots & d_{2,M} \\ \dots & \dots & \dots & \dots \\ d_{k,1} & d_{k,2} & \dots & d_{k,M} \end{pmatrix}$$

Probabilistic IR model

- Adding weights to each term

  $D_j = (t_{1,j}, t_{2,j}, \ldots, t_{N,j})$

  $t_{i,j} = TRW_i * f_{i,j}$

- Term Relevance Weight

  $TRW_i = \log (p_i \times (1-q_i) / q_i \times (1-p_i))$

  p_i **Probability of the term appearing in relevant documents**

  q_i **Probability of the term appearing in irrelevant documents**

Weighting schema in CodeBroker

$$sim(Q, D_j) = \sum_{i=1}^{T} (\log \frac{N - n_i + 0.5}{n_i + 0.5}) \frac{(k_1 + 1) tf_{i,j}}{K + tf_{i,j}} \frac{(k_3 + 1) qtf_i}{k_3 + qtf_i}$$

N is the number of components
n_i is the number of components whose documents contain the term *ti*
T is the number of terms in the component collection
$tf_{i,j}$ is the frequency of term *ti* in the document of the component *Dj*
qtf_i is the frequency of term *ti* in the query *Q*

$$K = k_1((1 - b) + b \cdot dl_j / avdl$$

k_1, k_3, b are empirically determined parameters depending on the nature of the document collection. In *CodeBroker*, k_1 is set to 1.2, k_3 to 1.0, and *b* to 0.75.
dl_j is the length of document D_j
avdl is the average length of all documents in the collection

Signature matching determines the constraint compatibility

- Reusable components must be compatible in signature
  - Signature is the syntactic interface of a module (method and class)
  - Improving the precision of retrieval
- Method level match
  - Exact match

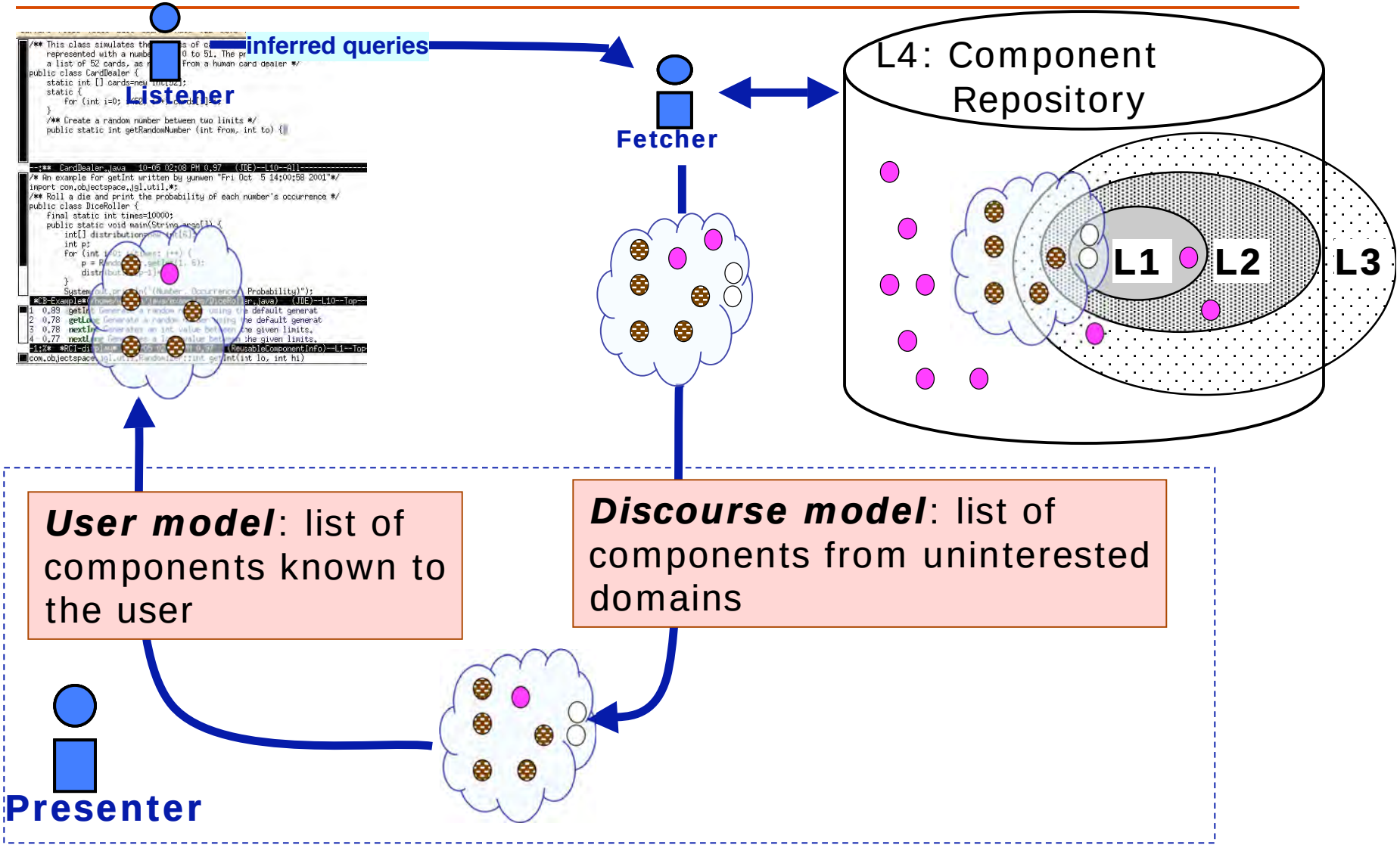
    `Type1 x Type2 -> Type3`

    `TypeA x TypeB -> TypeC`

    `<=> Type1=TypeA AND Type2=TypeB AND Type3=TypeC`
  - Relaxed match
    - Generalization / Specialization / Reorder

      string x int -> int **matches** (**relaxed**) long x string -> long

Presenter: tailoring the delivery to larger context and user



Discourse models: Improving task-relevance

- Discourse models capture the larger context of programming activities
  - Representing the interaction history between programmers and CodeBroker
  - Removing irrelevant components
  - Negative discourse models: specifying what is not of interest to programmers
  - Example:

```
((“java.util.zip”) ;; a package
 (“java.awt” (“CardLayout”))) ;; a
class
```

User models: User-specific delivery

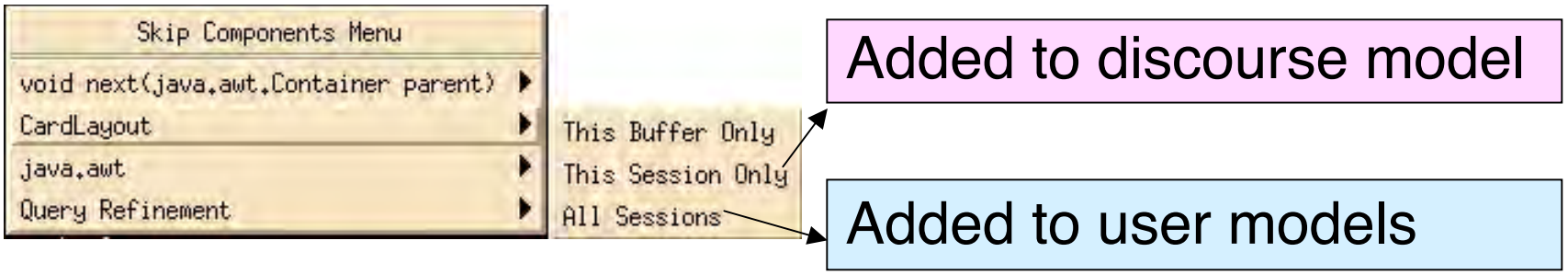
- User models represent programmers' knowledge on the component repository
  - A list of known components
  - Example:

```
((“java.applet” (“Applet” (“getParameterInfo”))
  (“java.io” (“File” (“exists”
                      “11/02/00” “11/10/00”
                       “11/11/00”)
                     (“isAbsolute”
                      “11/01/00” “11/10/00”
                      “11/11/00”))))
```

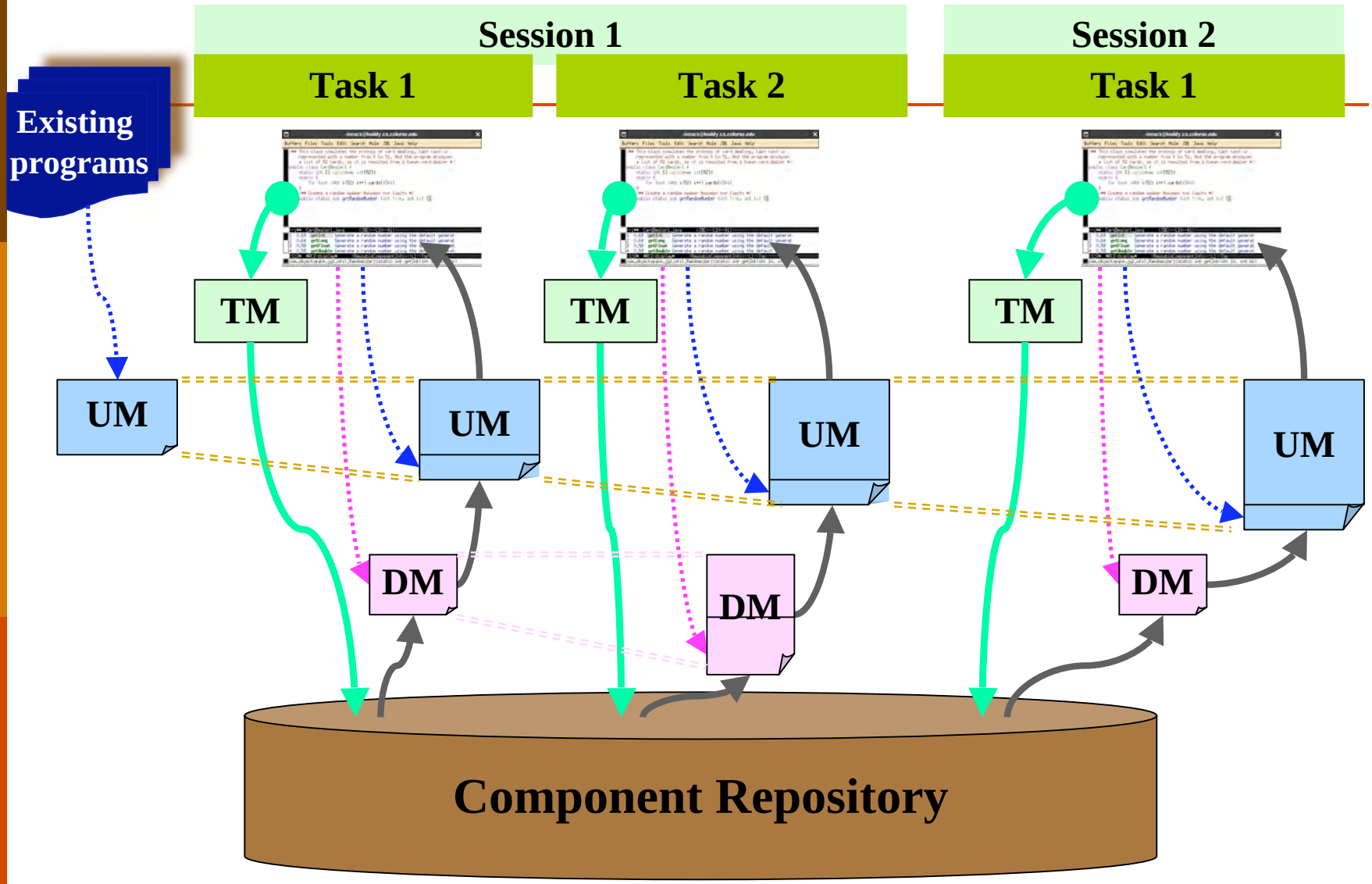
  - Components contained in user models are **not** delivered

Incremental discourse modeling and user modeling

- **Initial** user models
  - Created by analyzing existing user programs
- **Adaptive** user models
  - CodeBroker updates user models automatically when it detects the use of a component in the editor
- **Adaptable** user models and discourse models
  - Using the `Skip Components Menu` associated with each delivered component

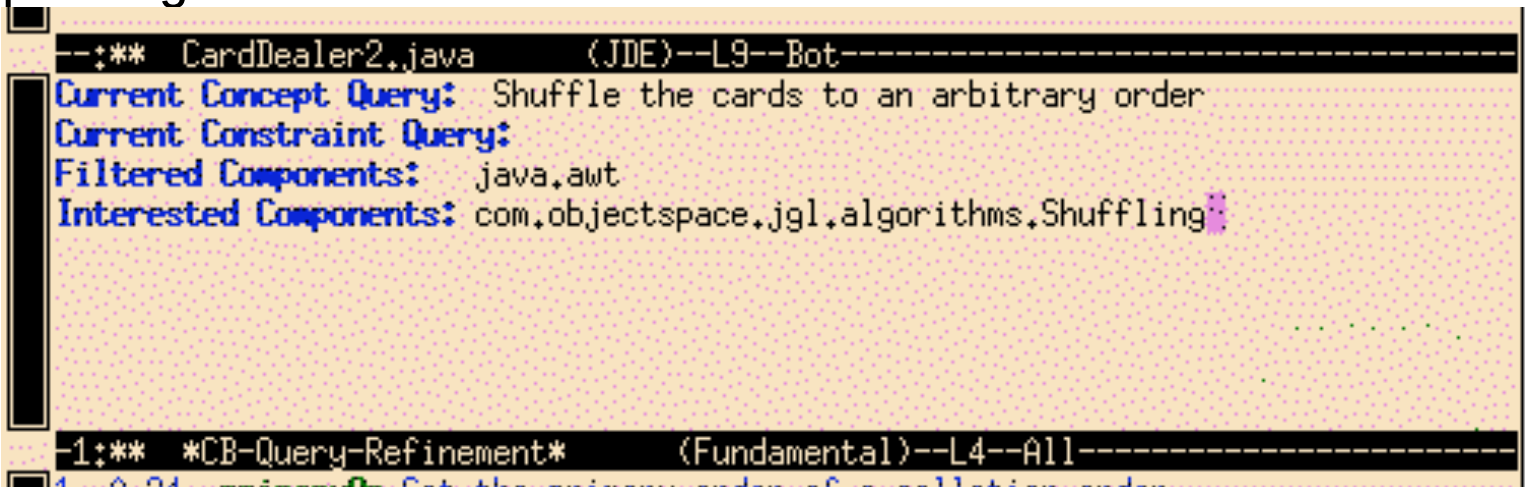


Models in CodeBroker



Retrieval-by-reformulation

- A process for software developers to incrementally develop reuse queries
- Delivered components help developers become familiar with the vocabulary and structure of the repository
  - Change the way of writing the query
  - Limit the search scope by specifying (un)interested packages and classes



The cycle of delivery-browsing-searching

- Delivered components are results of information reconnaissance
- Possible actions after the delivery
  - The needed component is delivered
    → Choose the needed one through browsing
  - Too many components are delivered
    → Filter the delivered components
  - The needed one is not delivered
    → Search again through retrieval-by-reformulation

Evaluation experiments

- Experiment goals:
  - Observe the effectiveness of CodeBroker in encouraging programmers to reuse
  - Analyze the effectiveness of task inference, discourse models, and user models
- 12 experiments with 5 subjects
  - Implementing an assigned task with CodeBroker

Subjects	S1	S2	S3	S4	S5
Years of prog. in general	3-4	5-6	8	10+	10+
Java skill (self-evaluation)	4	7	7-8	10	7

System assessment

Sub	No	total	delivered	breakdown of deliveries			triggered
				unanticipated (L4-L3)	anticipated but unknown (L3)	vaguely known (L2)	
S1	1	10	4	2	2	0	0
	2	3	1	1	0	0	1
S2	3	7	1	1	0	0	0
	4	4	1	1	0	0	0
	5	5	3	0	2	1	1
S3	6	5	2	1	1	0	1
	7	4	3	1	2	0	1
	8	3	0	0	0	0	0
S4	9	4	3	0	3	0	0
	10	3	1	1	0	0	2
S5	11	4	1	1	0	0	2
	12	5	0	0	0	0	0
Sum		**57**	**20**	**9**	**10**	**1**	**8**

The STeP_IN system: a SocioTechnical Platform for in situ Networking

Three proximities for knowledge collaboration

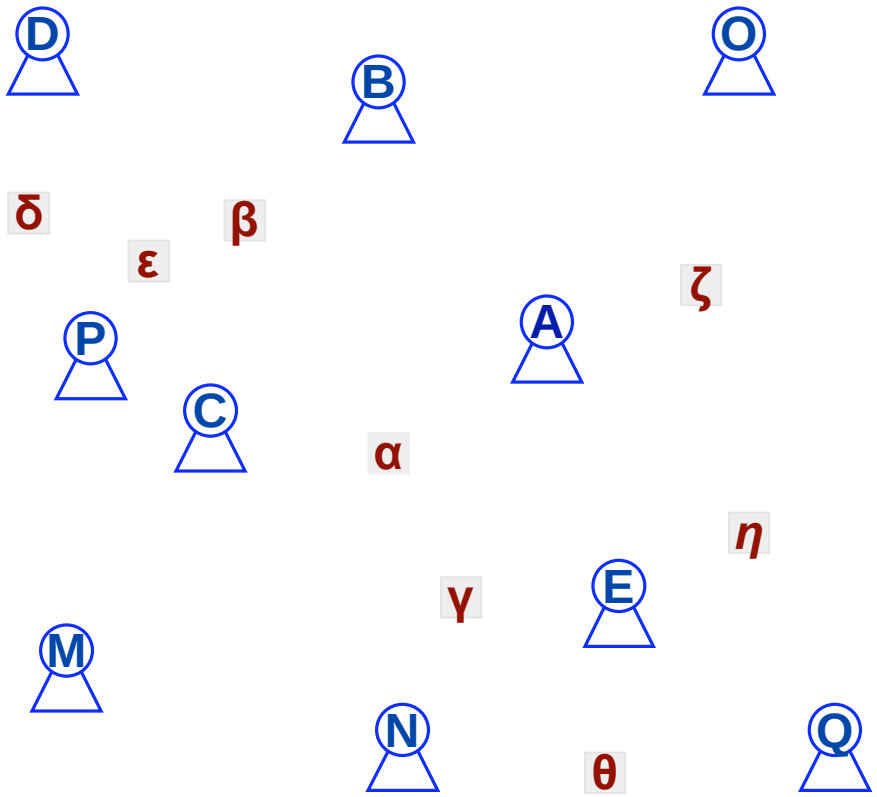
- Cognitive proximity
  - Defines the transferability and combinability of knowledge
- Structural proximity
  - Provides communication channels for knowledge to flow
- Relational proximity
  - Determines the motivation to participate in knowledge collaboration

Dynamic community

- A dynamic community is a small group of knowledge workers that forms *ad hoc* in support of a particular *user* working on a particular *task,* and dissembles as the task is finished

- Dynamic communities support *situated* knowledge collaboration by mobilizing positive forces in *all three proximities*

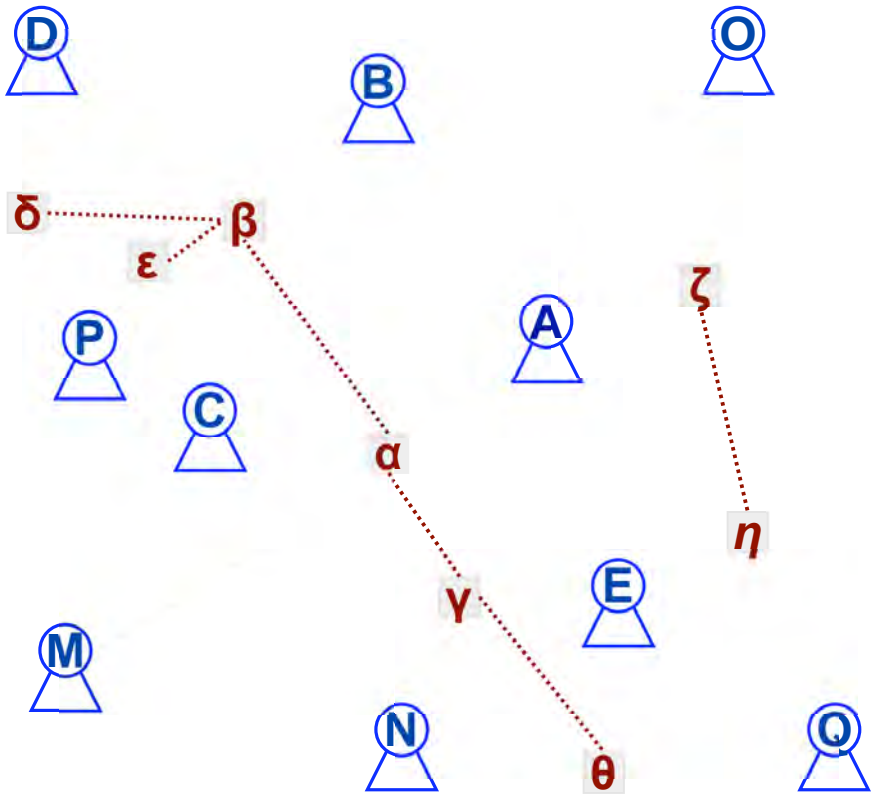
Knowledge Work Space



Set of people $\Psi = \{A, B, C, D, E, M, N, O, P, Q\}$

Set of information $\Phi = \{\alpha, \beta, \gamma, \delta, \varepsilon, \zeta, \eta, \theta\}$

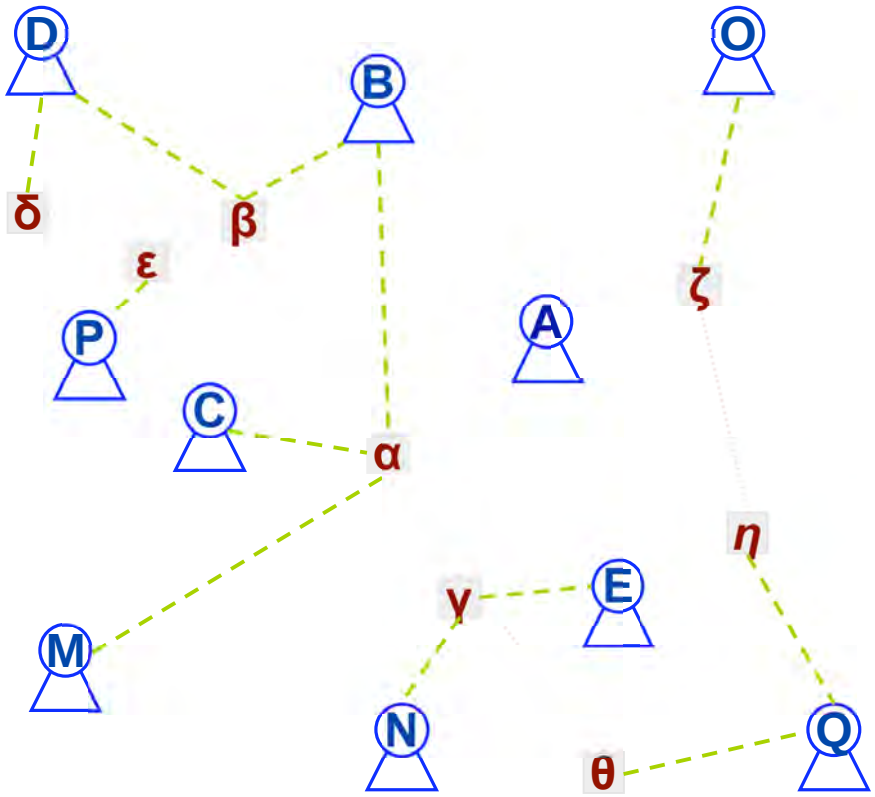
Knowledge Work Space



Relation between information

$\Pi = \{ (\alpha, \beta), (\alpha, \gamma), (\beta, \varepsilon), (\beta, \delta), (\gamma, \theta), (\zeta, \eta) \}$

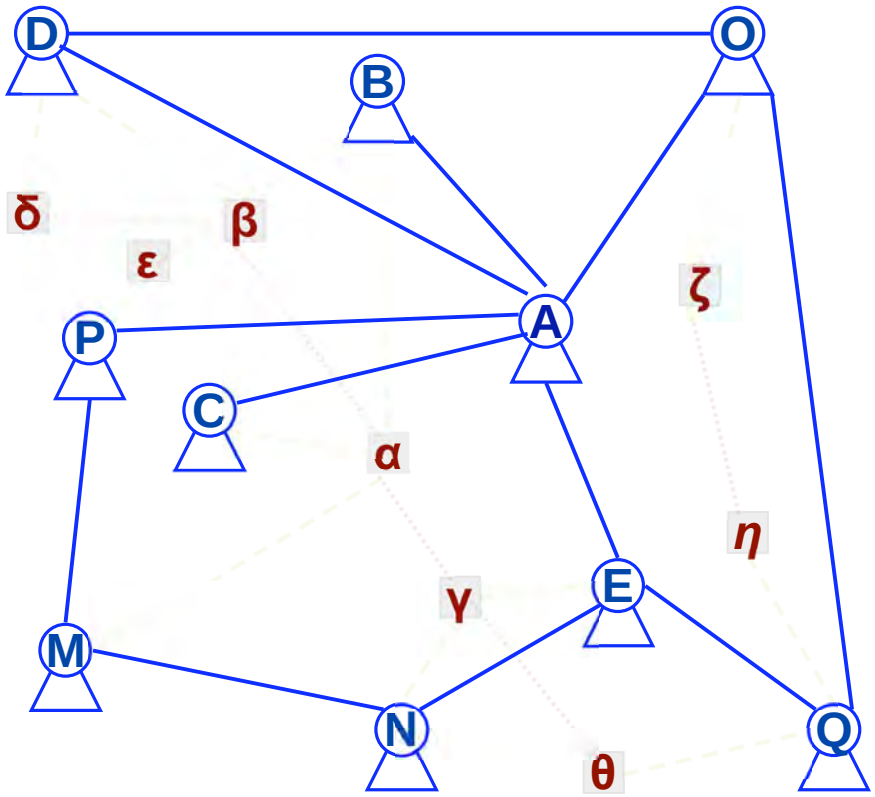
Knowledge Work Space



Relation between people and information

PI = { (B, α), (C, α), (M, α), (B, β), (D, β), (E, γ),
(N, γ), (D, δ), (P, ε), (O, ζ), (Q, η), (Q, θ)}

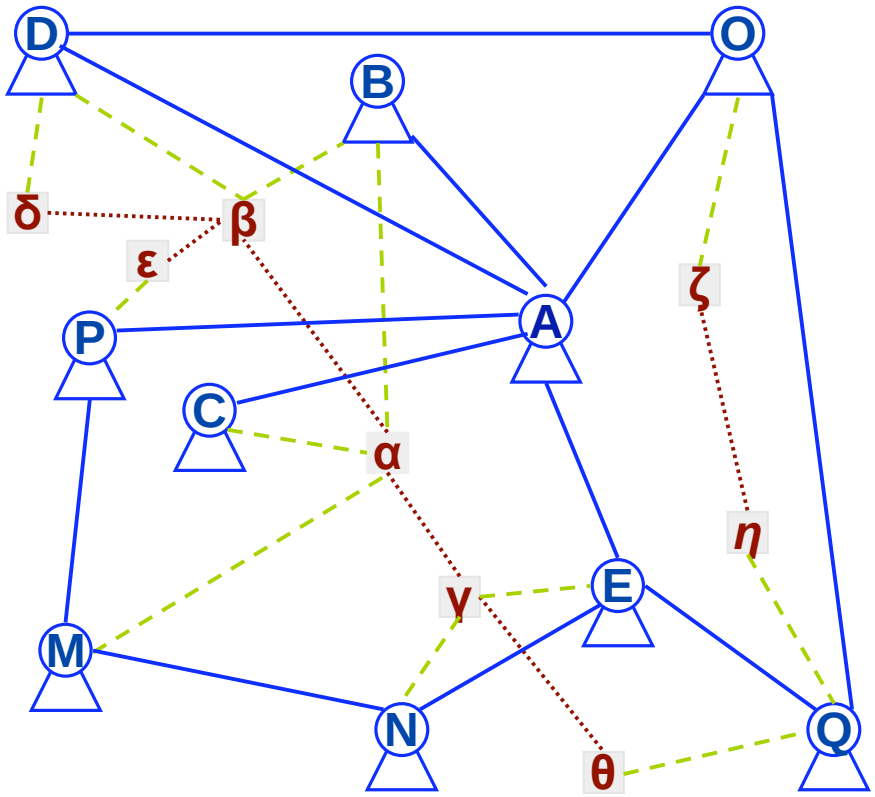
Knowledge Work Space



Relation between people

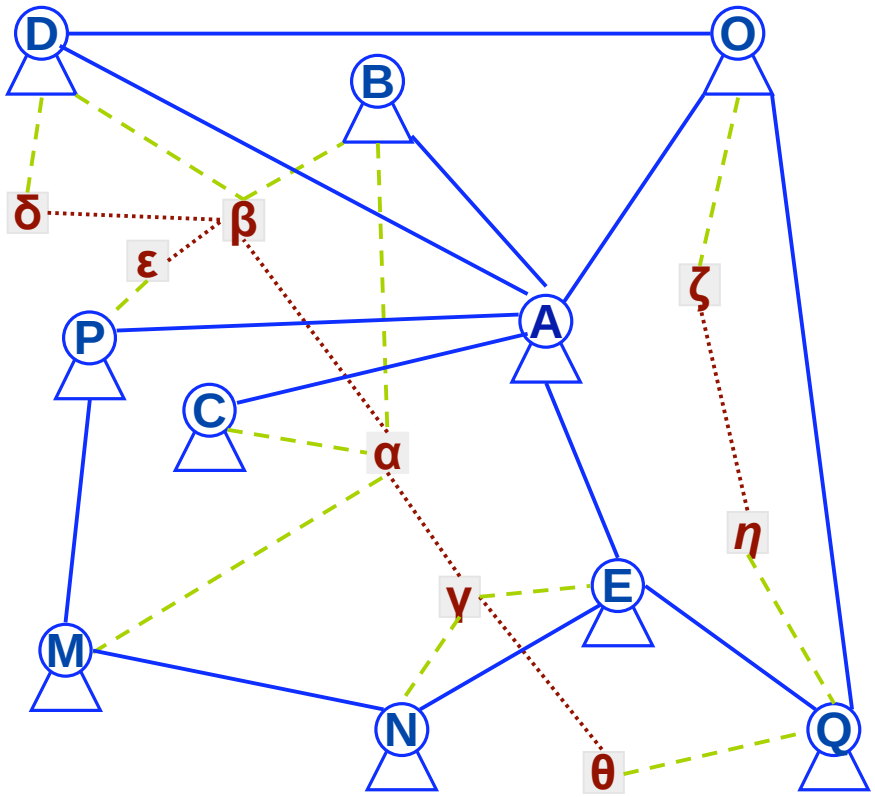
PP = { (A, B), (A, C), (A, D), (A, E), (A, O), (A, P),
(D, O), (E, N), (E, Q), (M, P), (M, N), (O, Q) }

Knowledge Work Space

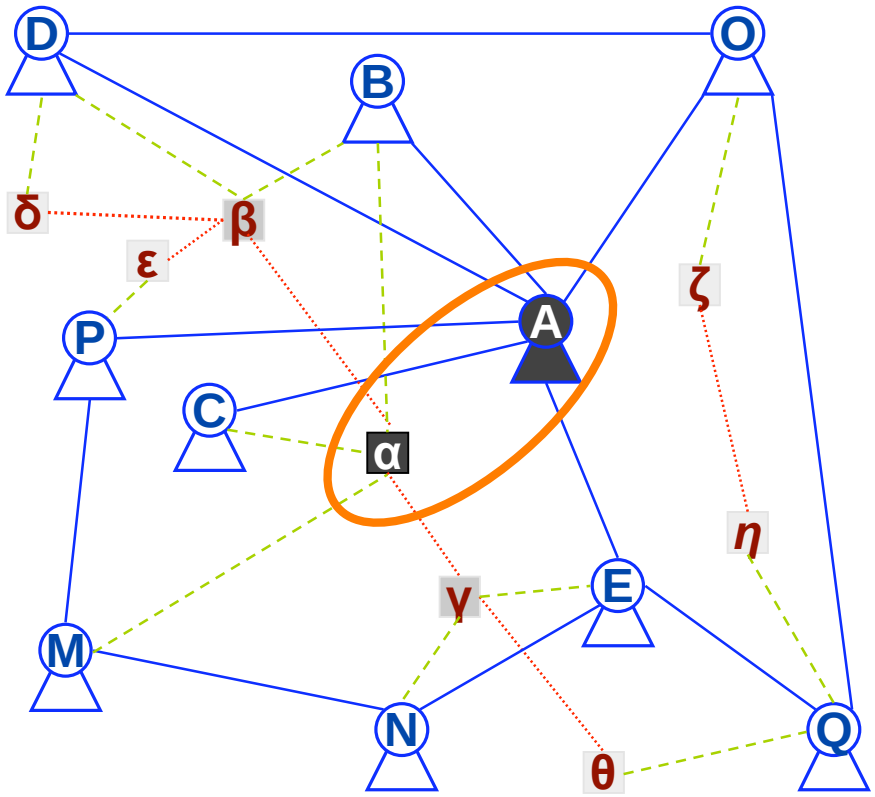


KWS= ({(α,β), (α,γ), (β,ε), (β,δ), (γ,θ), (ζ,η)},
{(B,α), (C,α), (M,α), (B,β), (D,β), (E,γ), (N,γ), (D,δ), (P,ε), (O,ζ), (Q,η), (Q,θ)},
{(A,B), (A,C), (A,D), (A,E), (A,O), (A,P), (D,O), (E,N), (E,Q), (M,P), (M,N), (O,Q)})

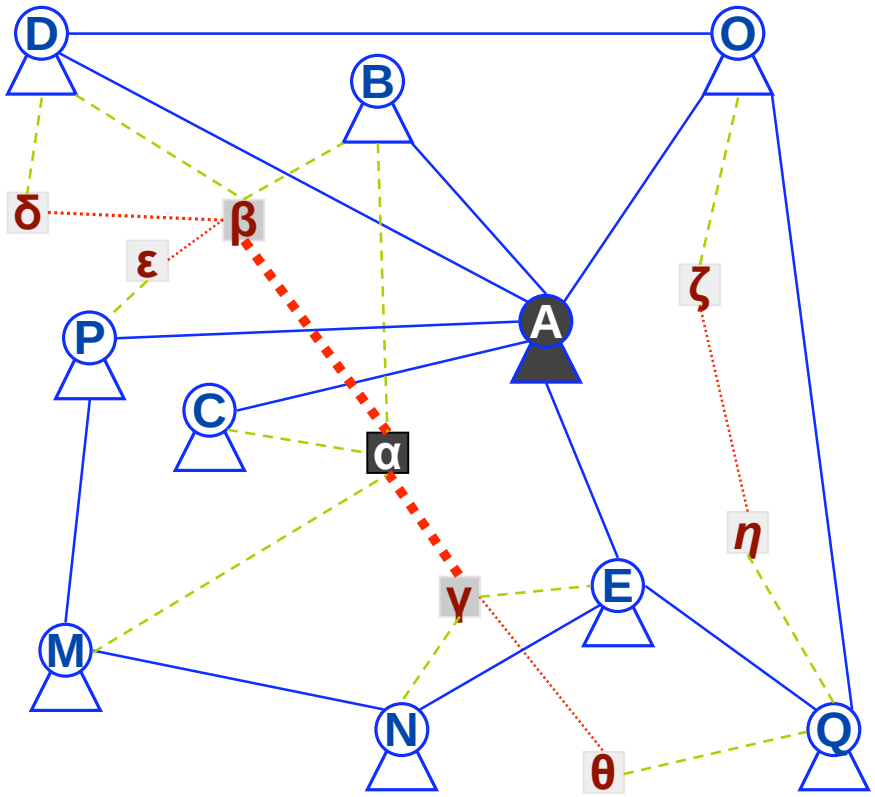
The forming process of a DynC



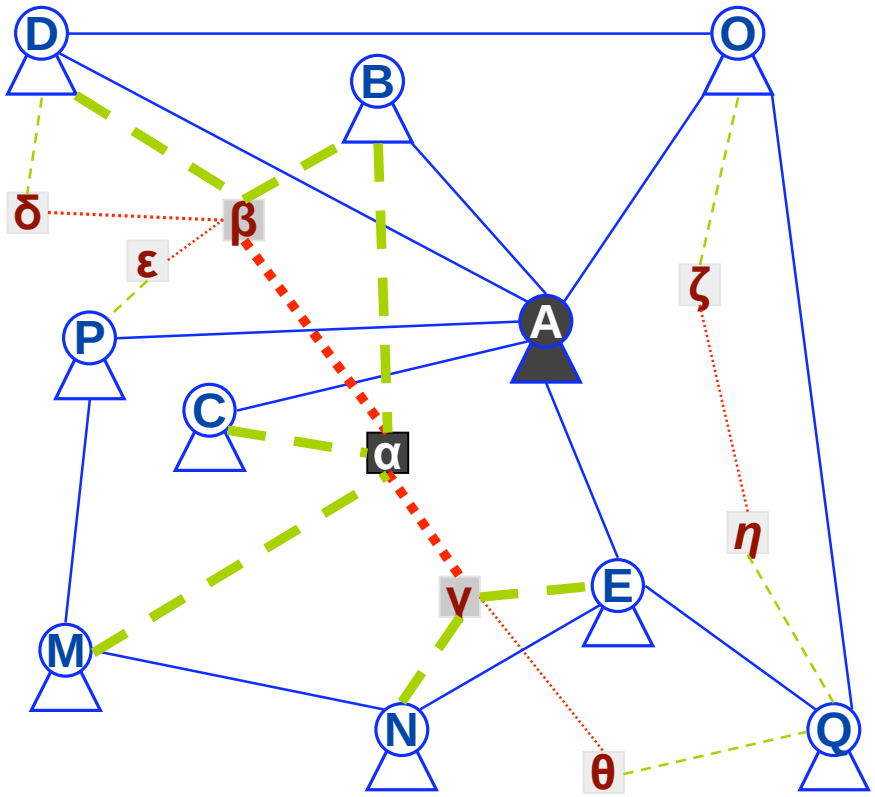
Triggering event for *Dync(A, α)*



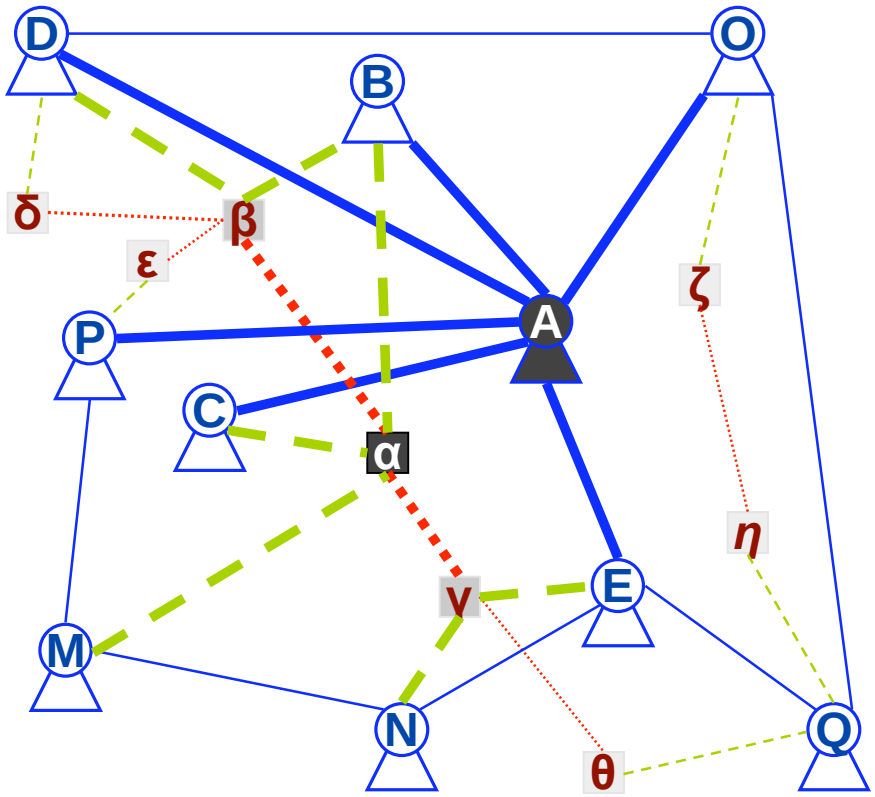
From information to information



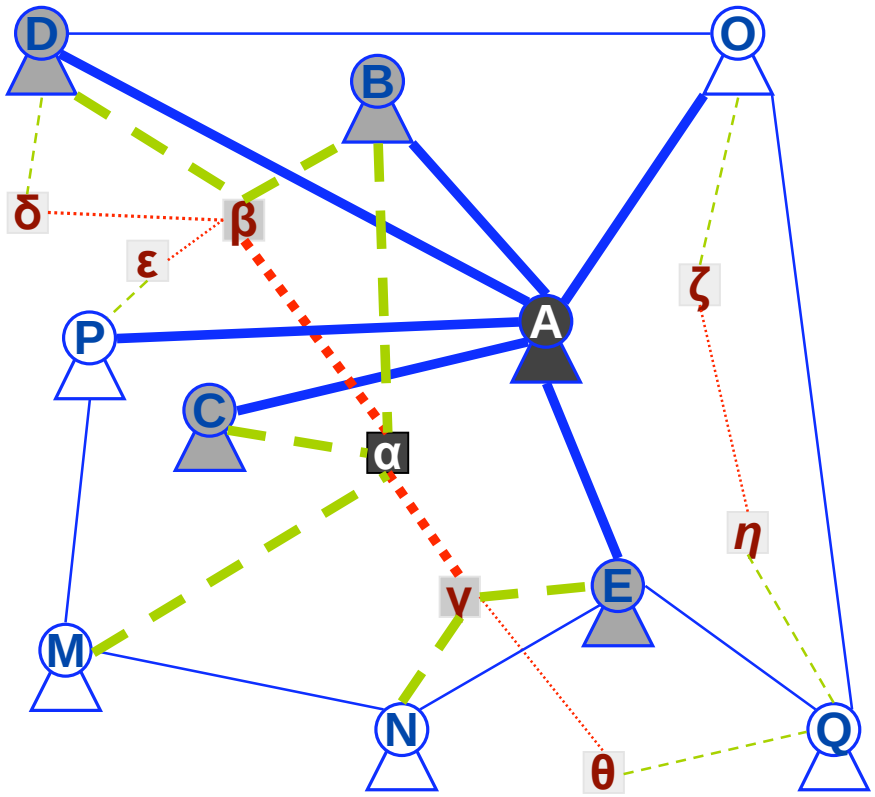
From information to people (experts)



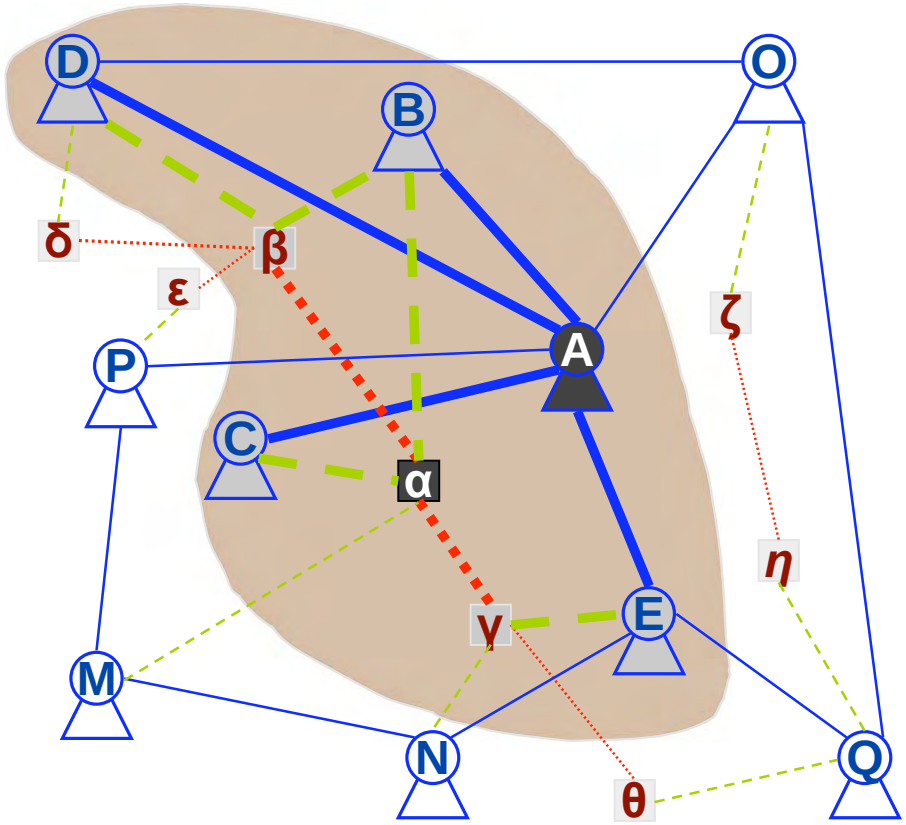
From people to people



$Dync(A, \alpha) = \{A, B, C, D, E\}$



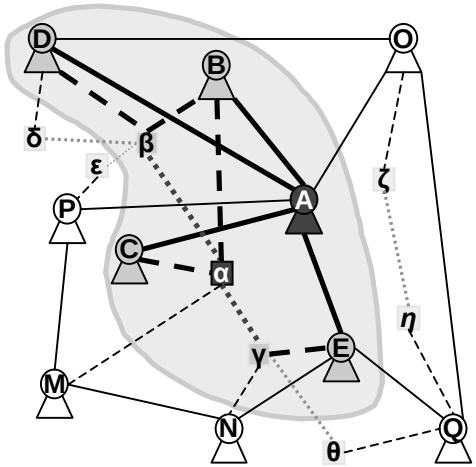
$Dync(A, \alpha) = \{A, B, C, D, E\}$



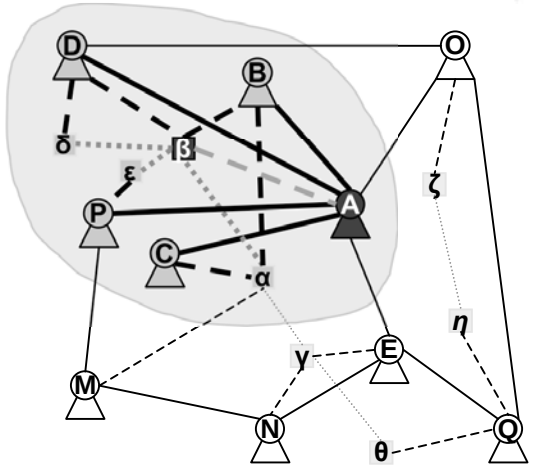
Task-specific and member-specific

Task-specific

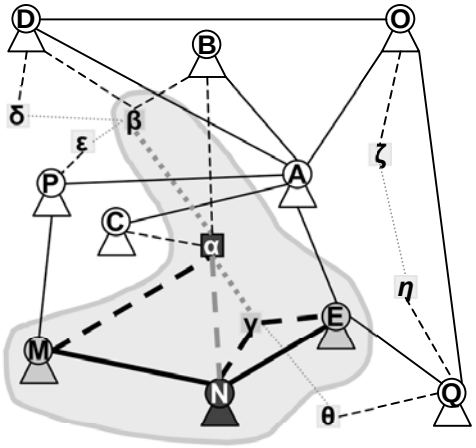
Member-specific



DynC(A, α) = {A, B, C, D, E}

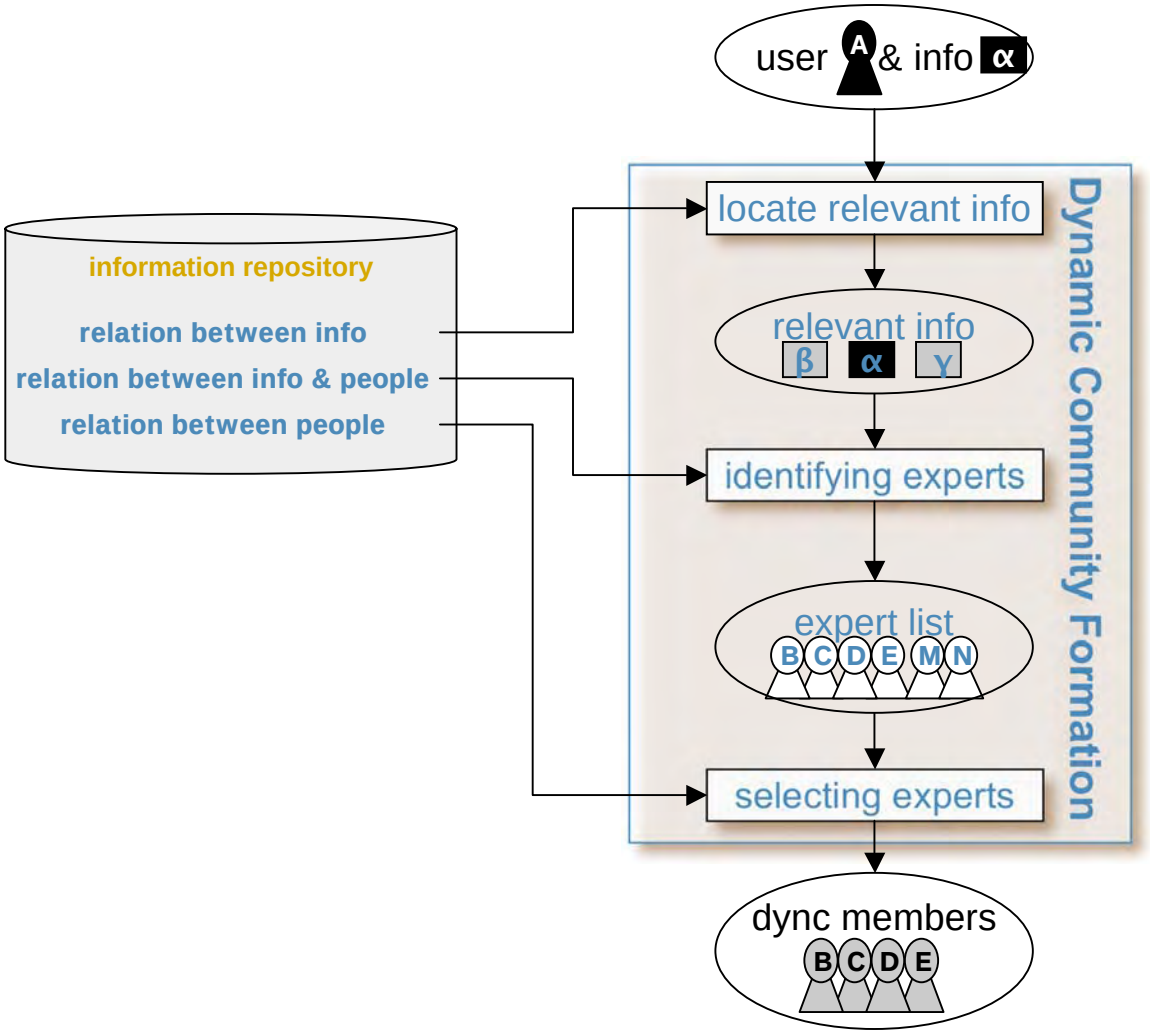


DynC(A, β) = {A, B, C, D, P}



DynC(N, α) = {E, N, M}

DynC formation support subsystem

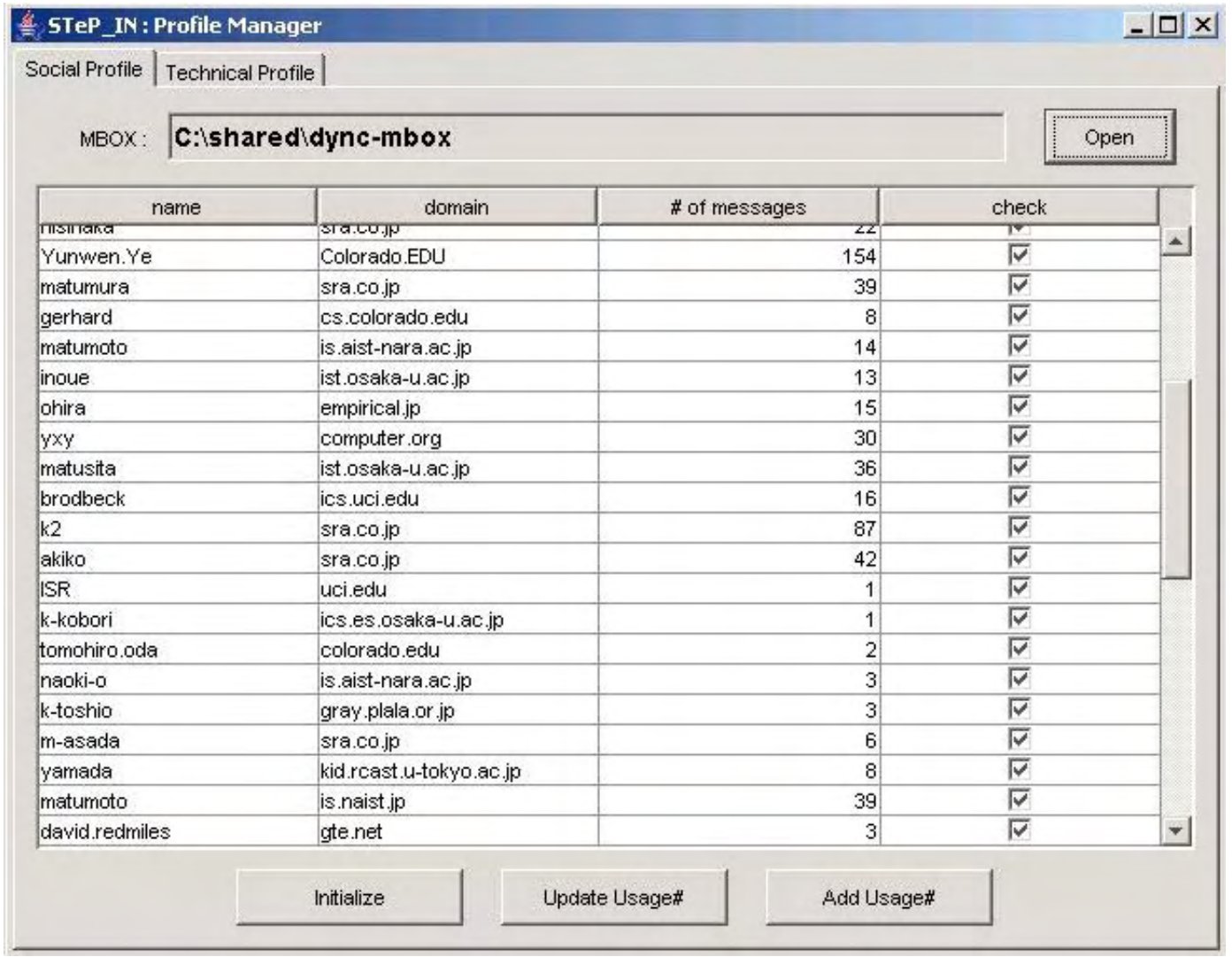


STeP_IN: programming with external knowledge resources

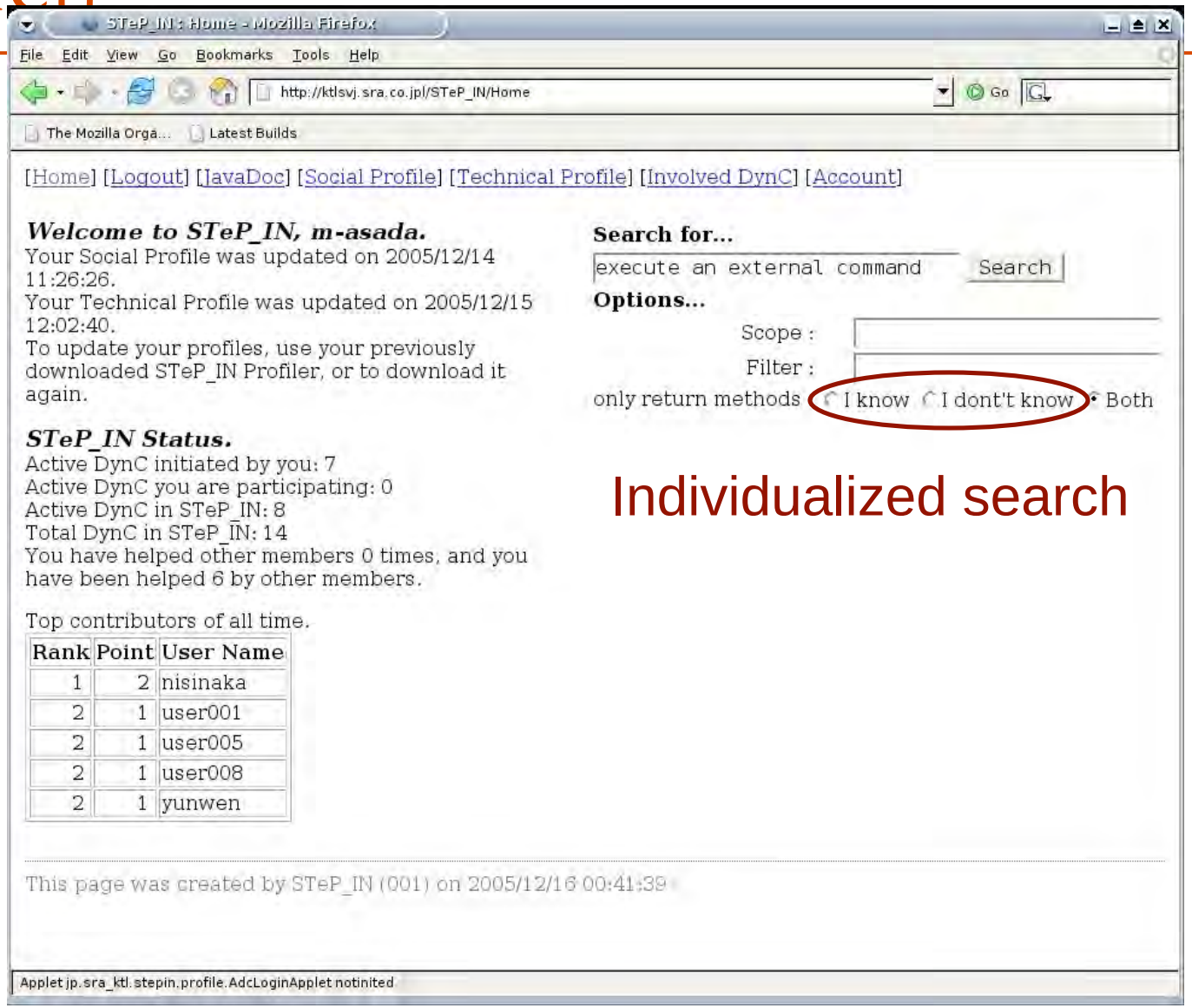
- Individualized search for API methods
- Accumulating and showing use examples
- Getting help from peers (a DynC approach)
  - Who are the experts?
    - Experts can only be identified after task is known
  - Who is willing to help?
    - Utilizing existing social relations

Technical Profile initialization

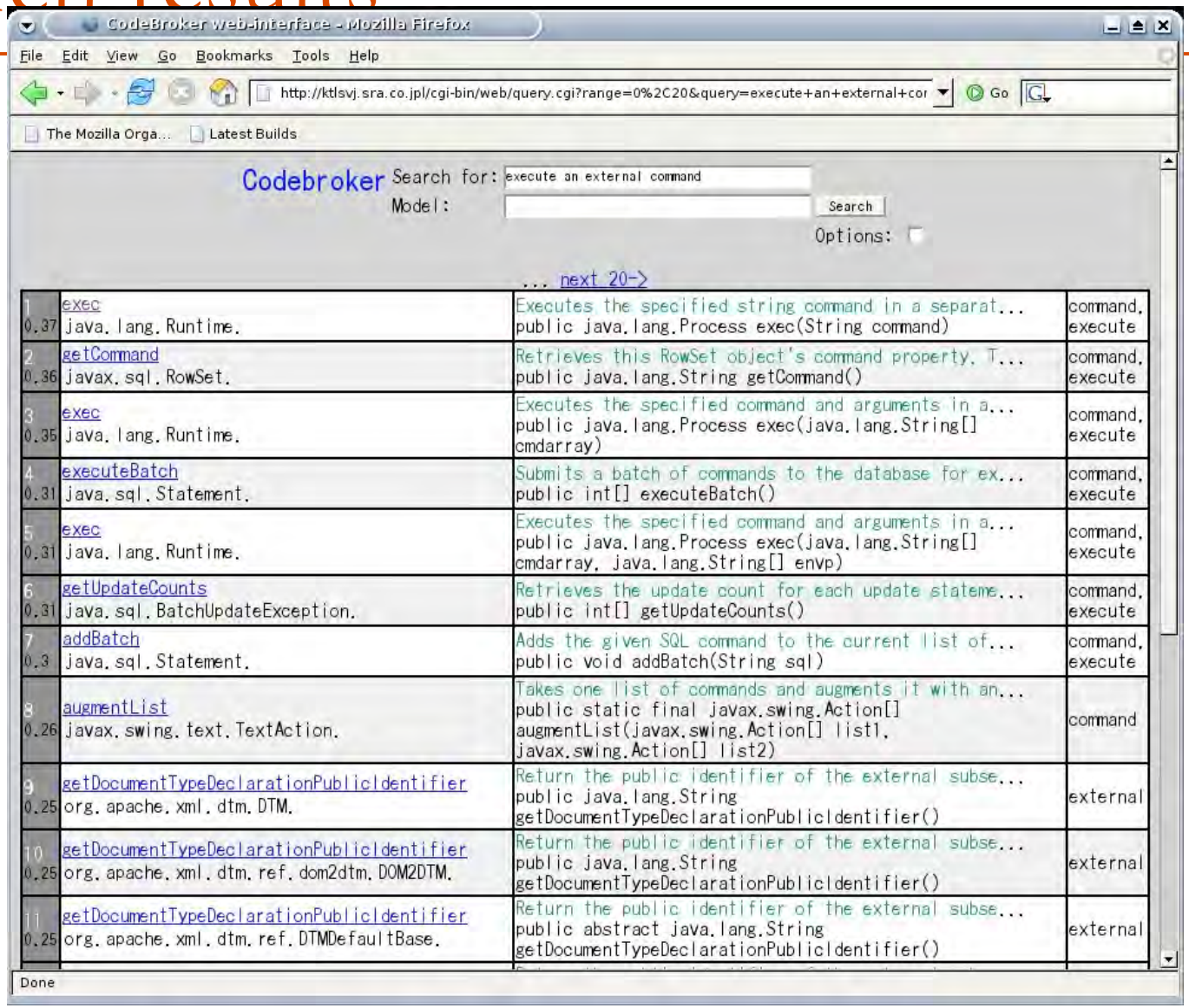
Social Profile Initialization



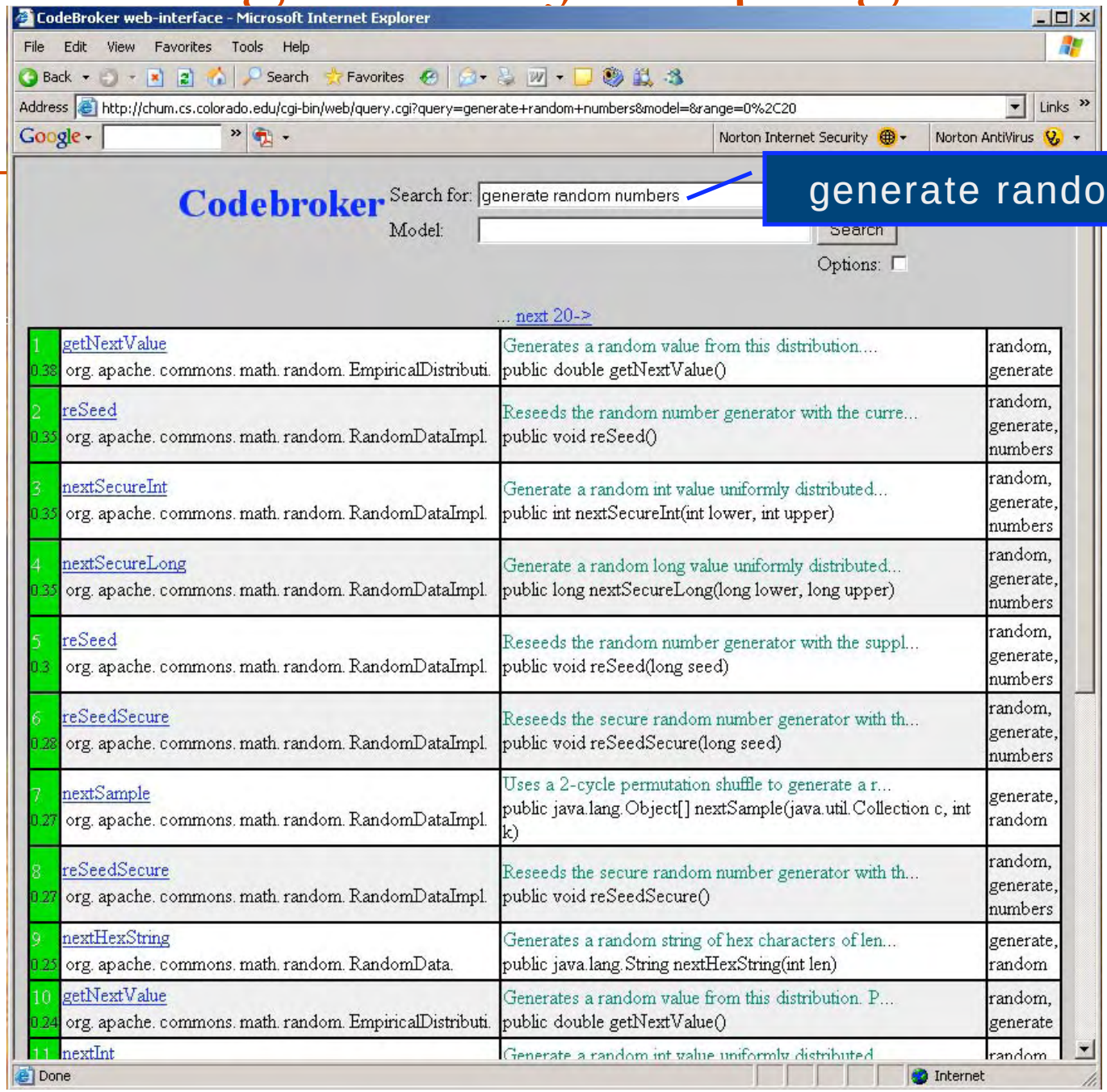
Search



Search results

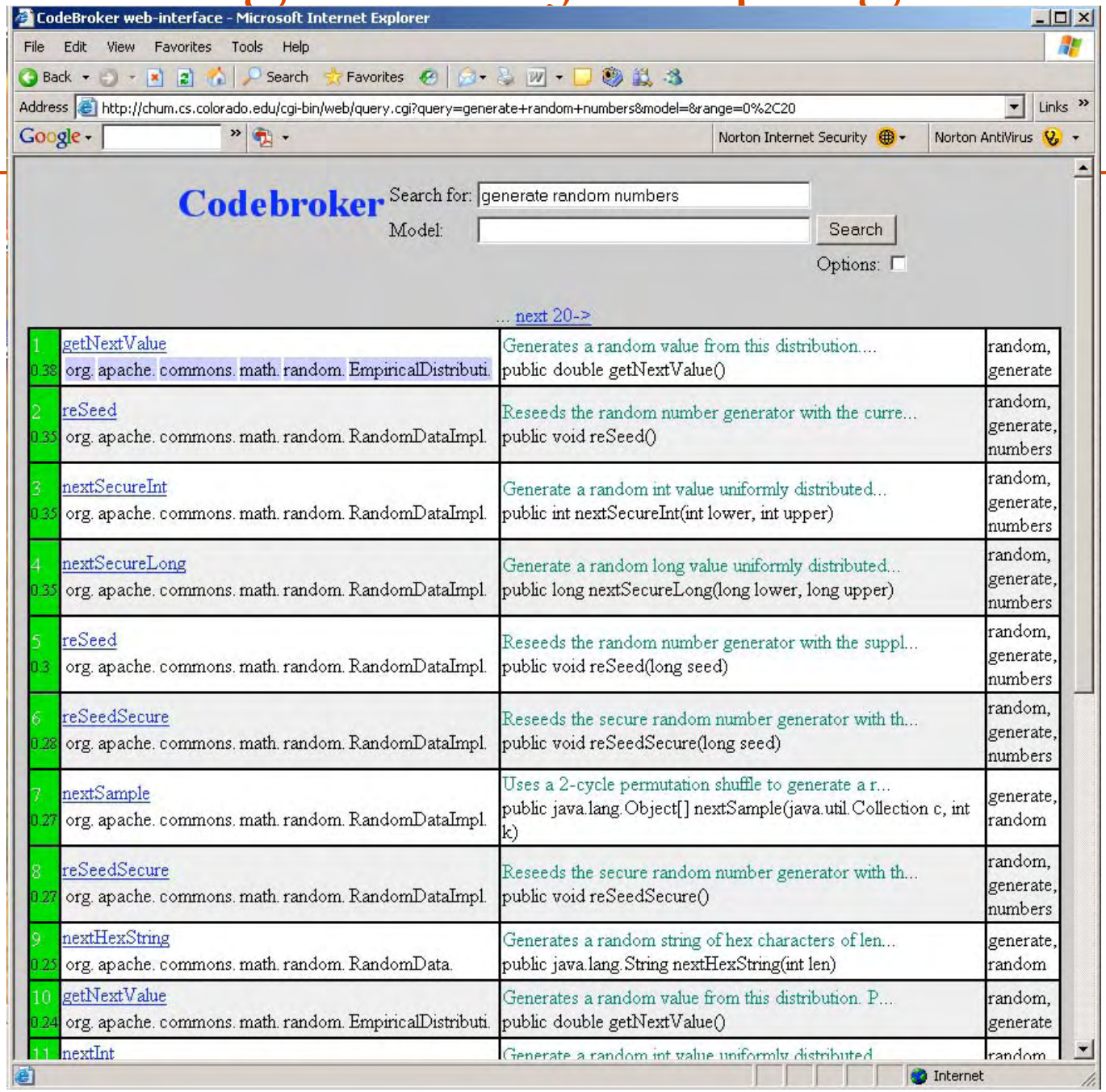


Search engine for Java programs

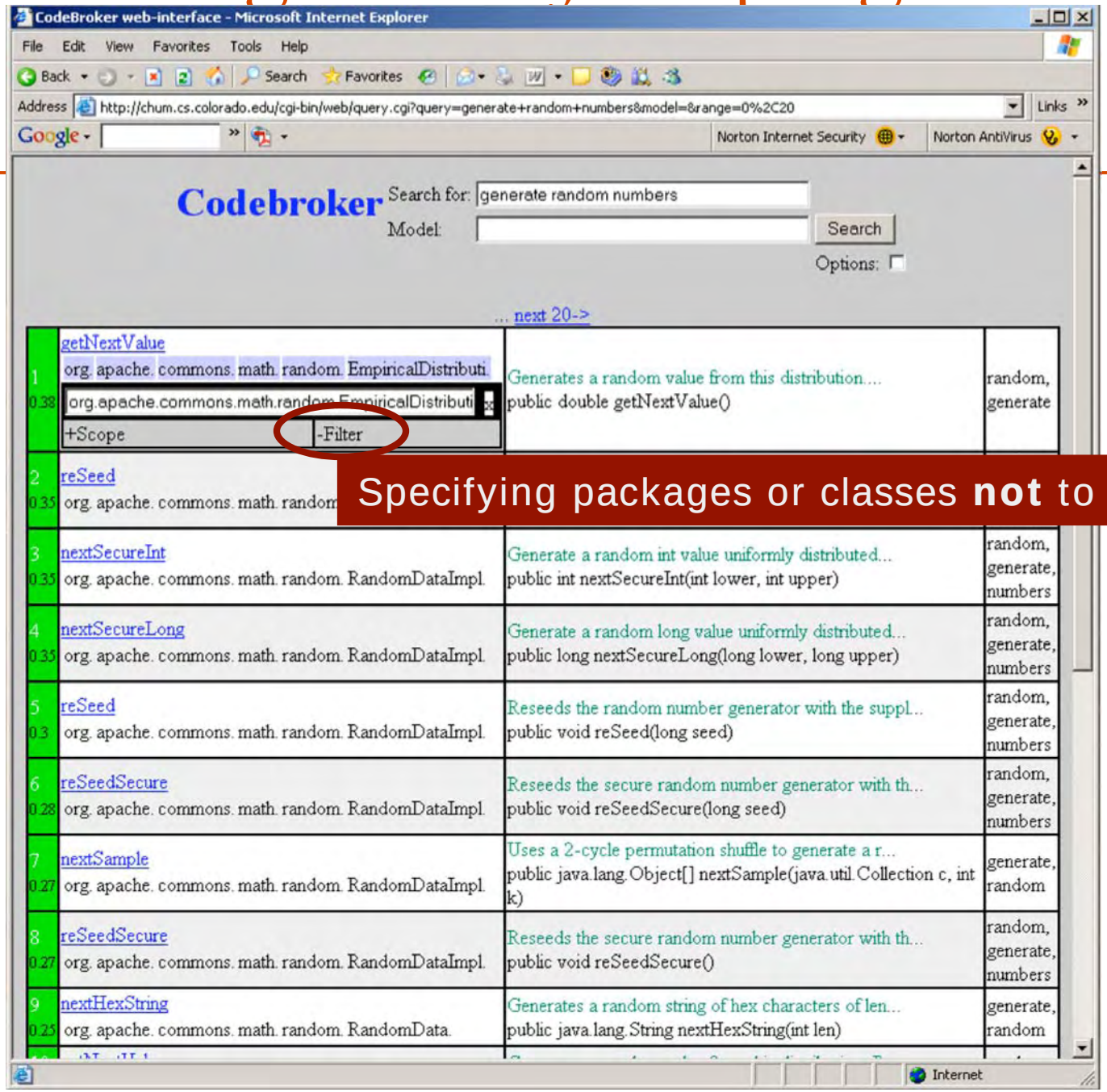


generate random numbers

Search engine for Java programs

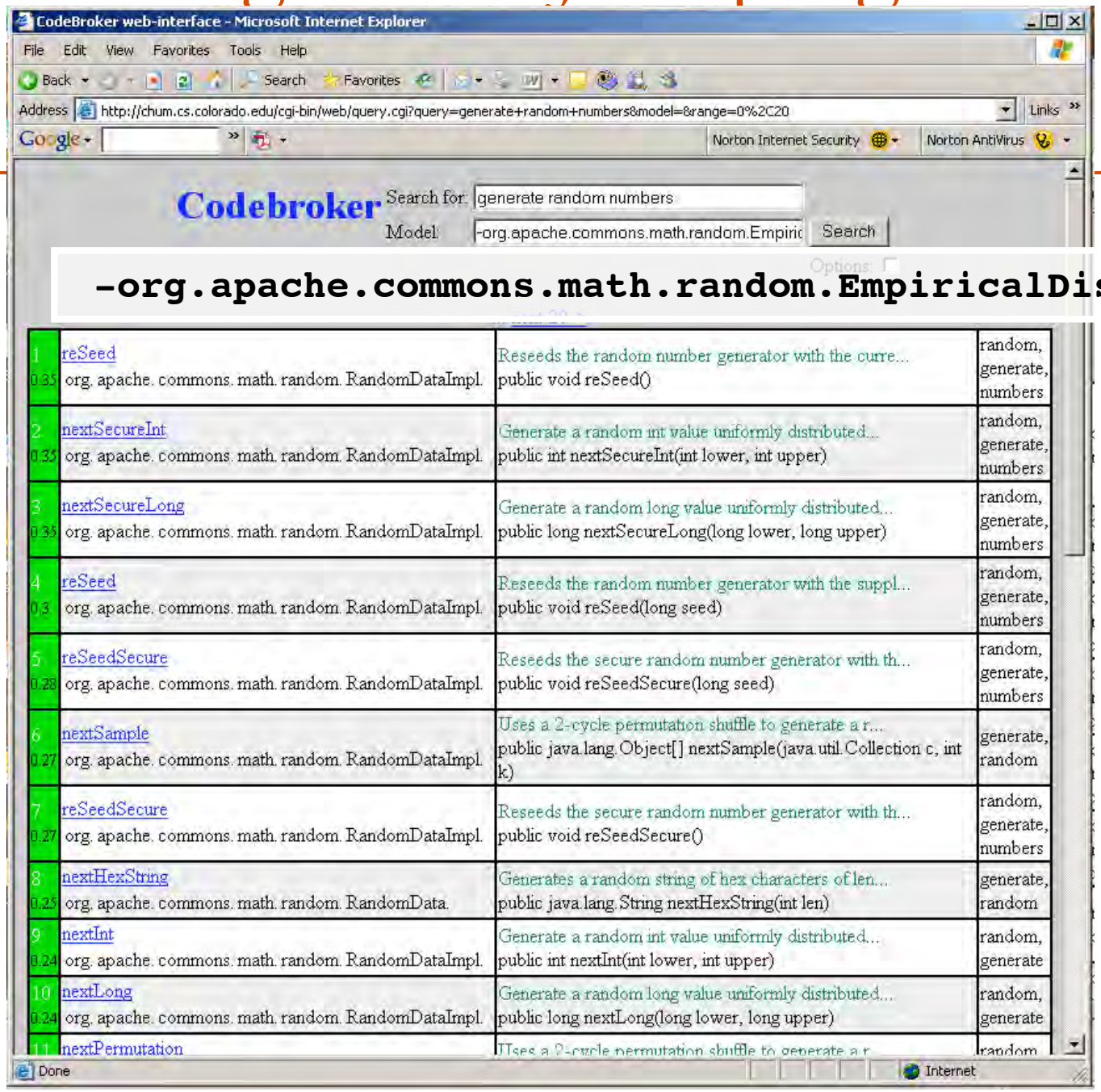


Search engine for Java programs

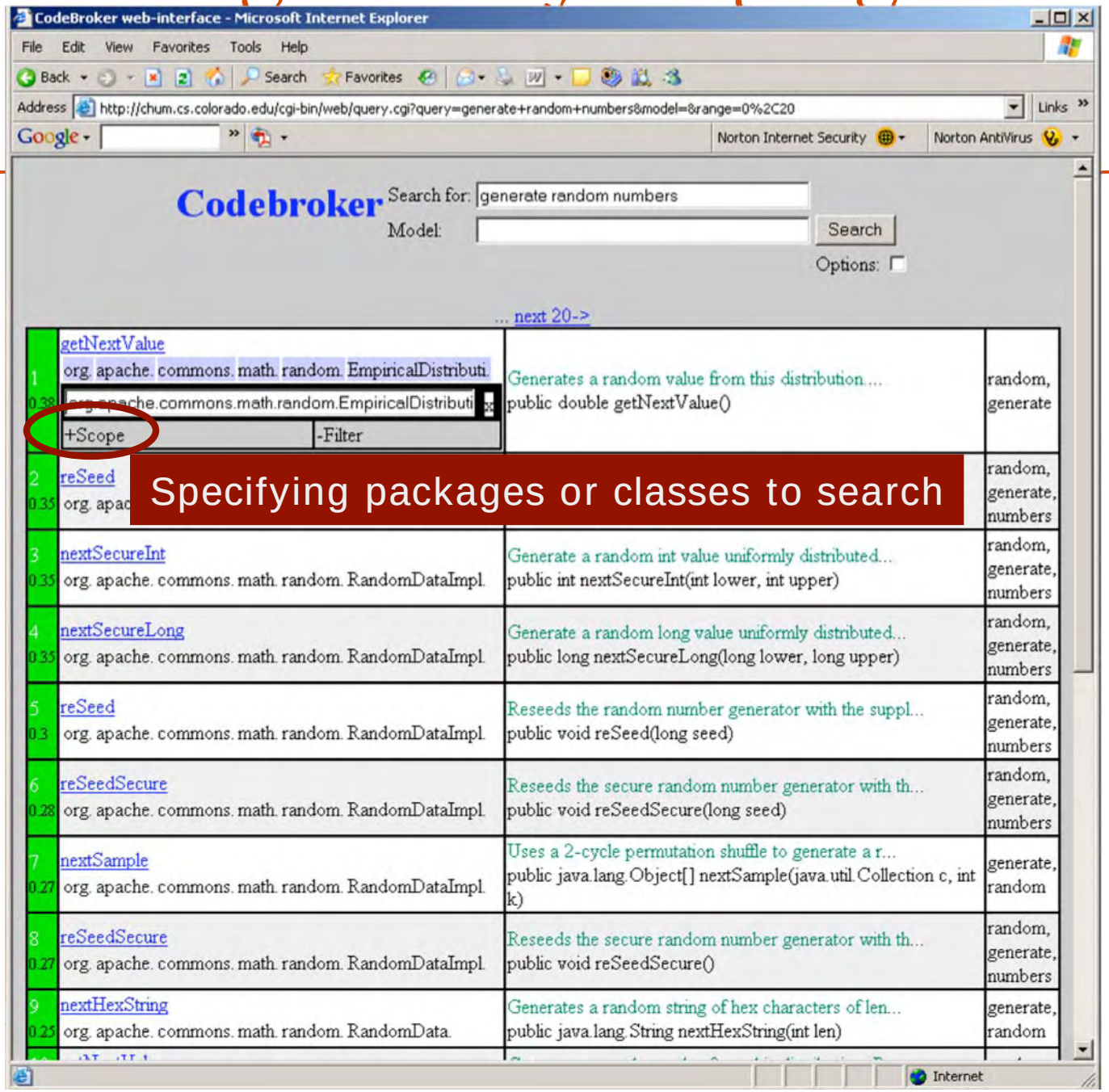


Specifying packages or classes **not** to search

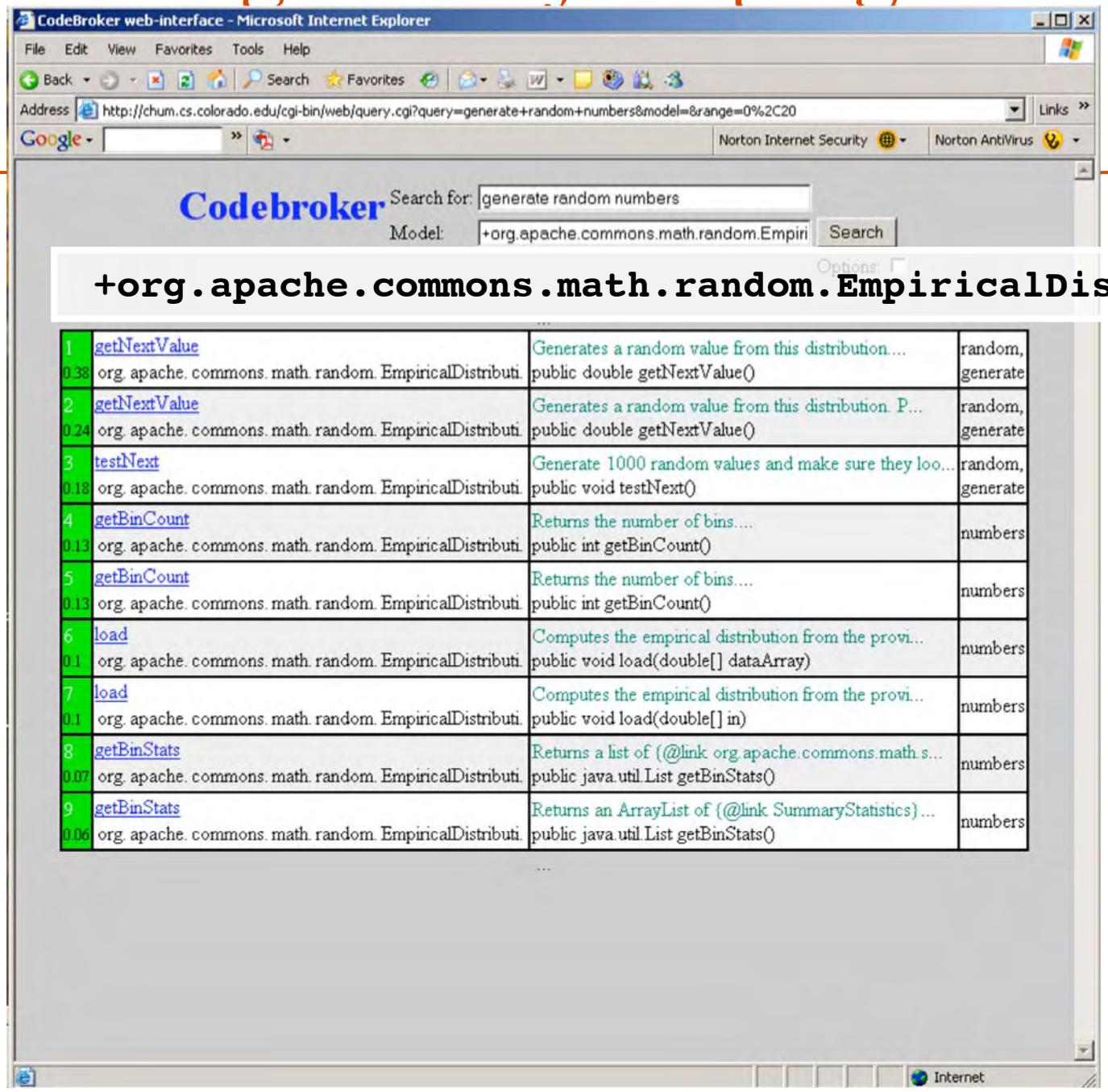
Search engine for Java programs



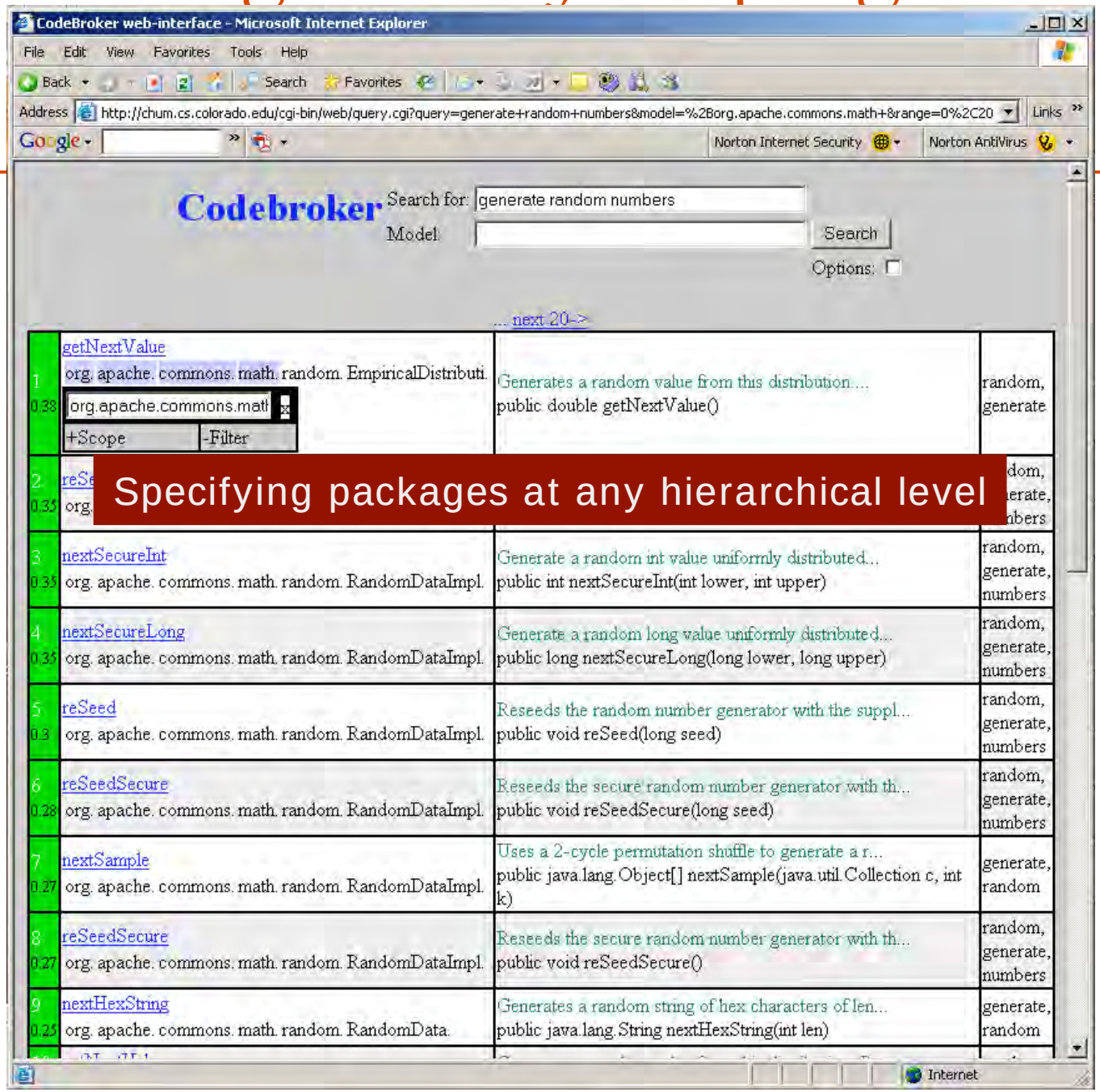
Search engine for Java programs



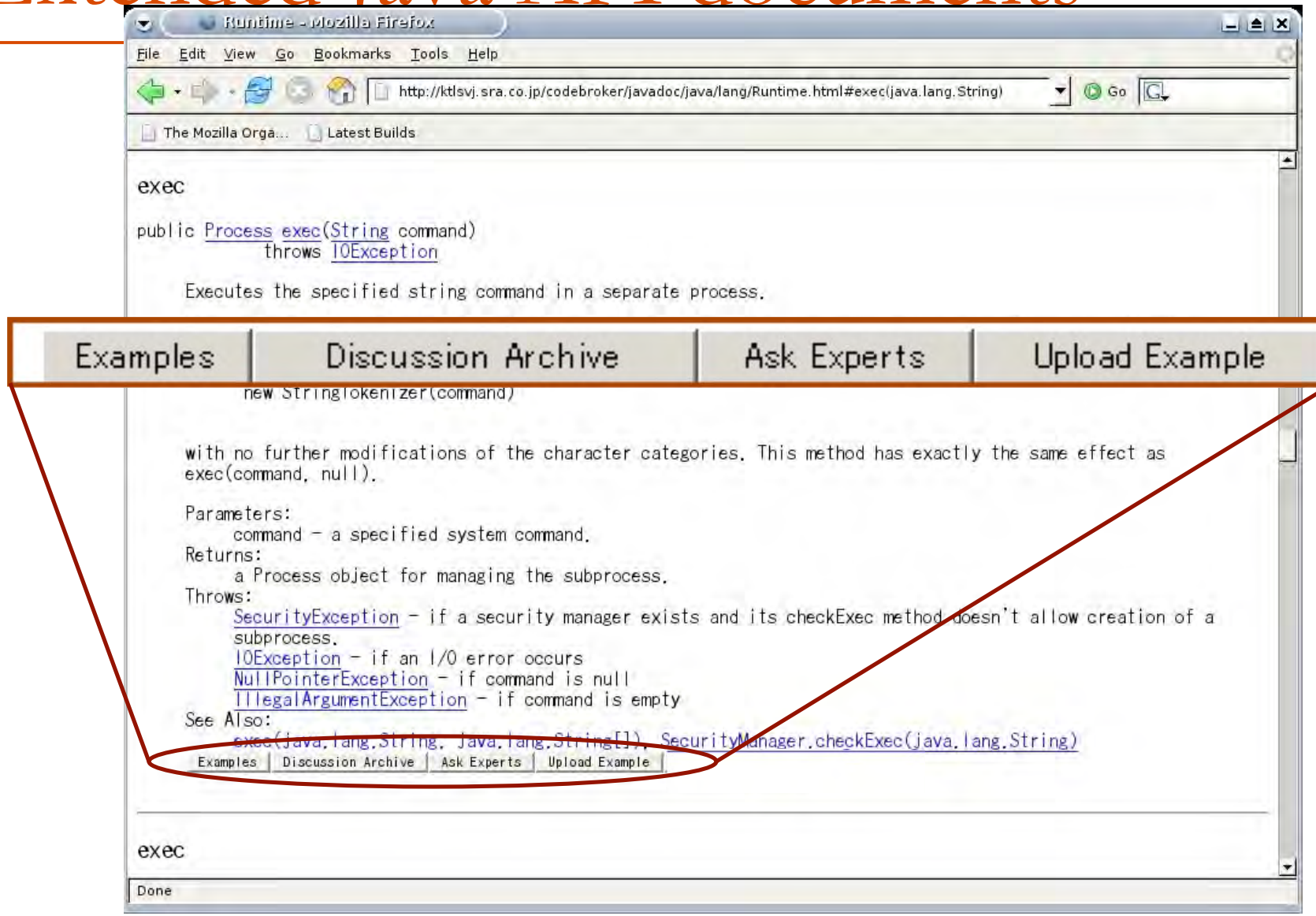
Search engine for Java programs



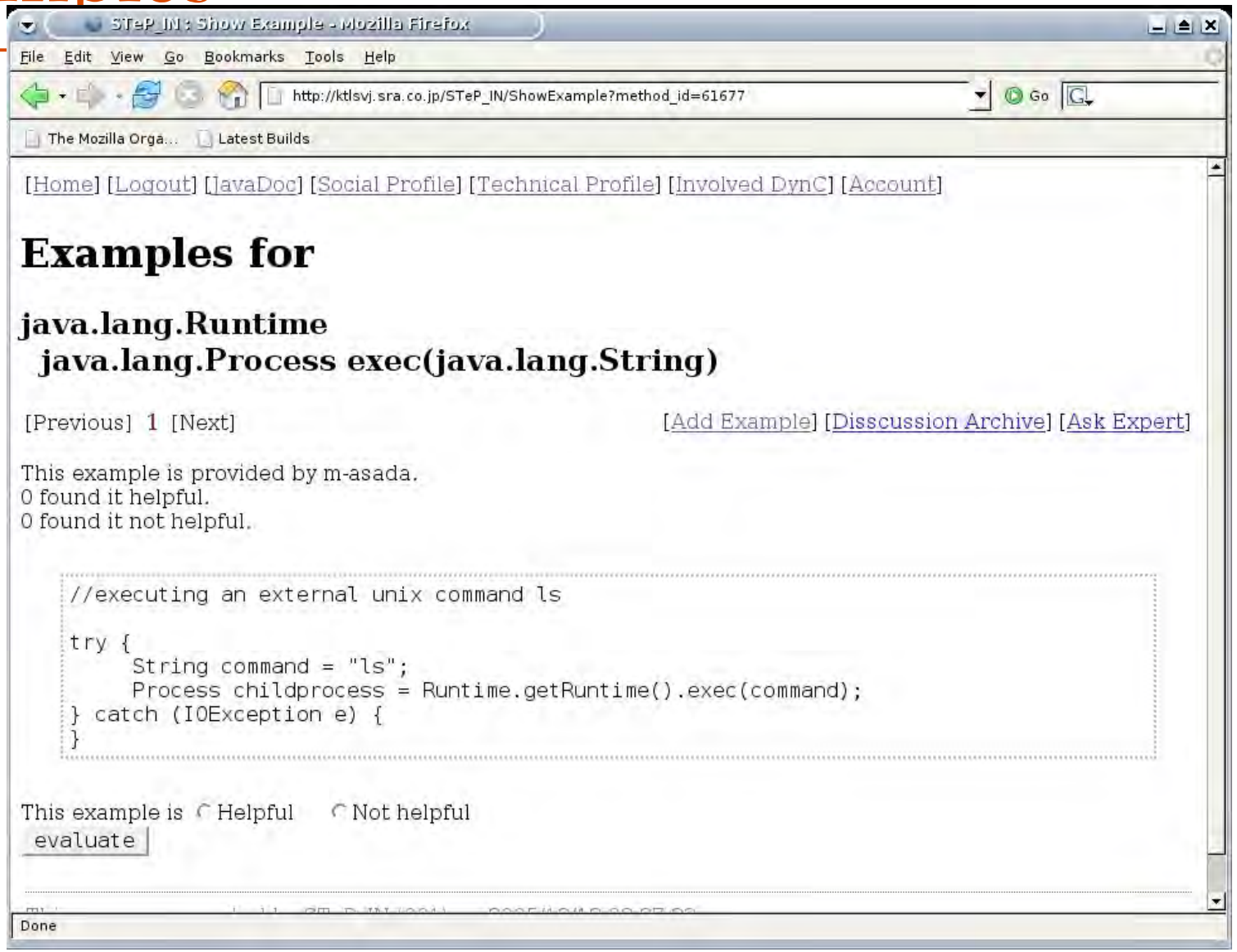
Search engine for Java programs



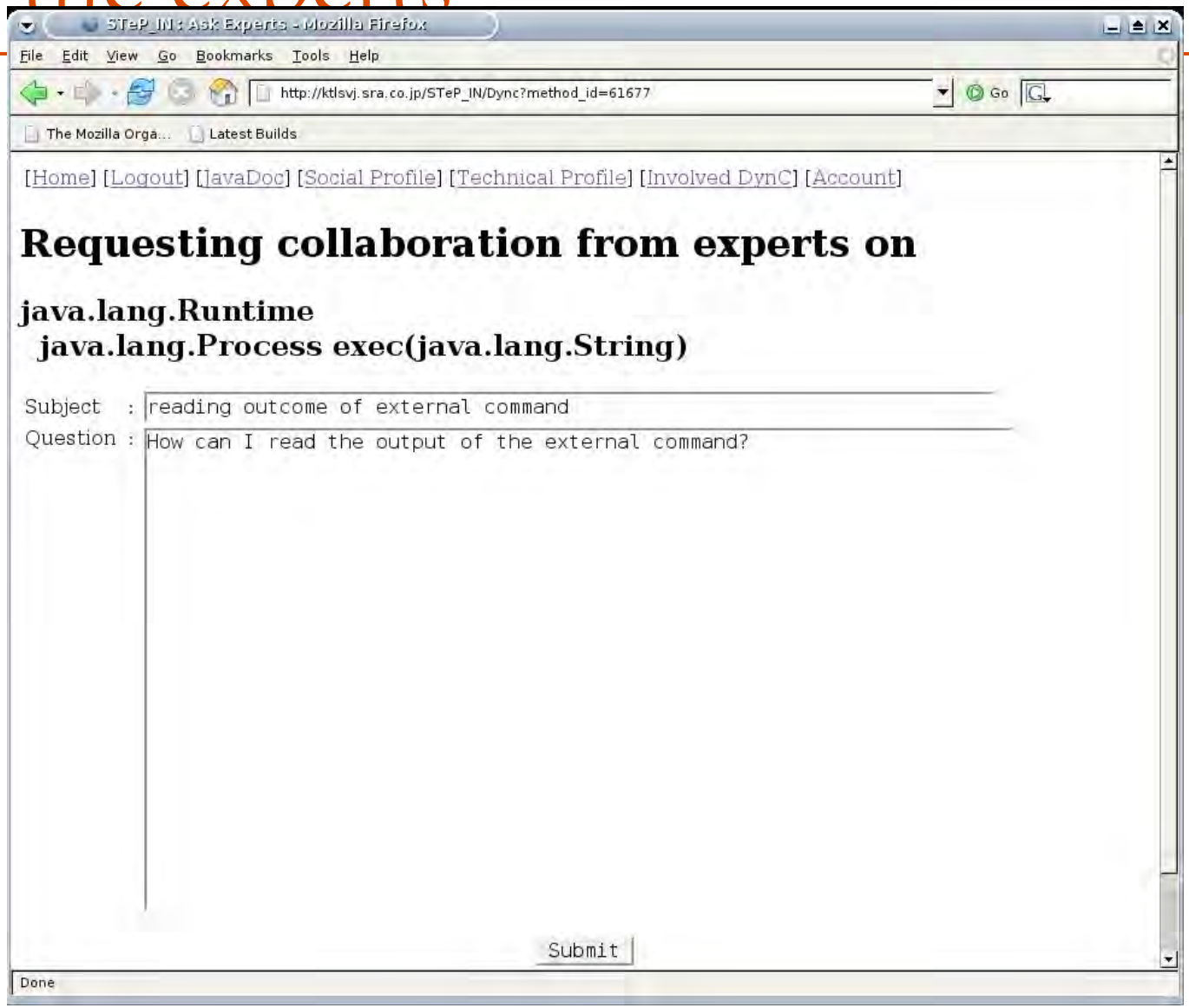
Extended Java API documents



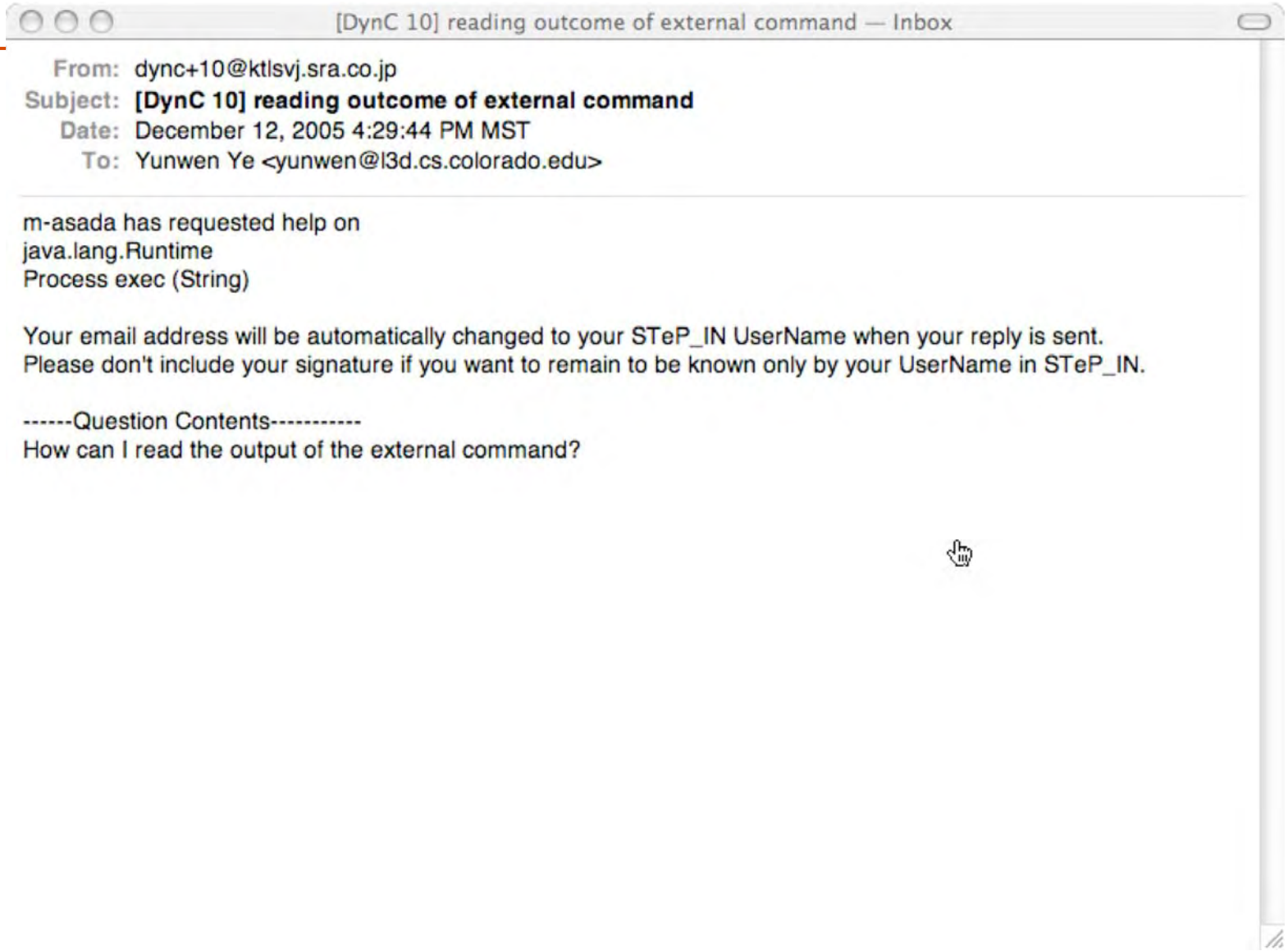
Examples



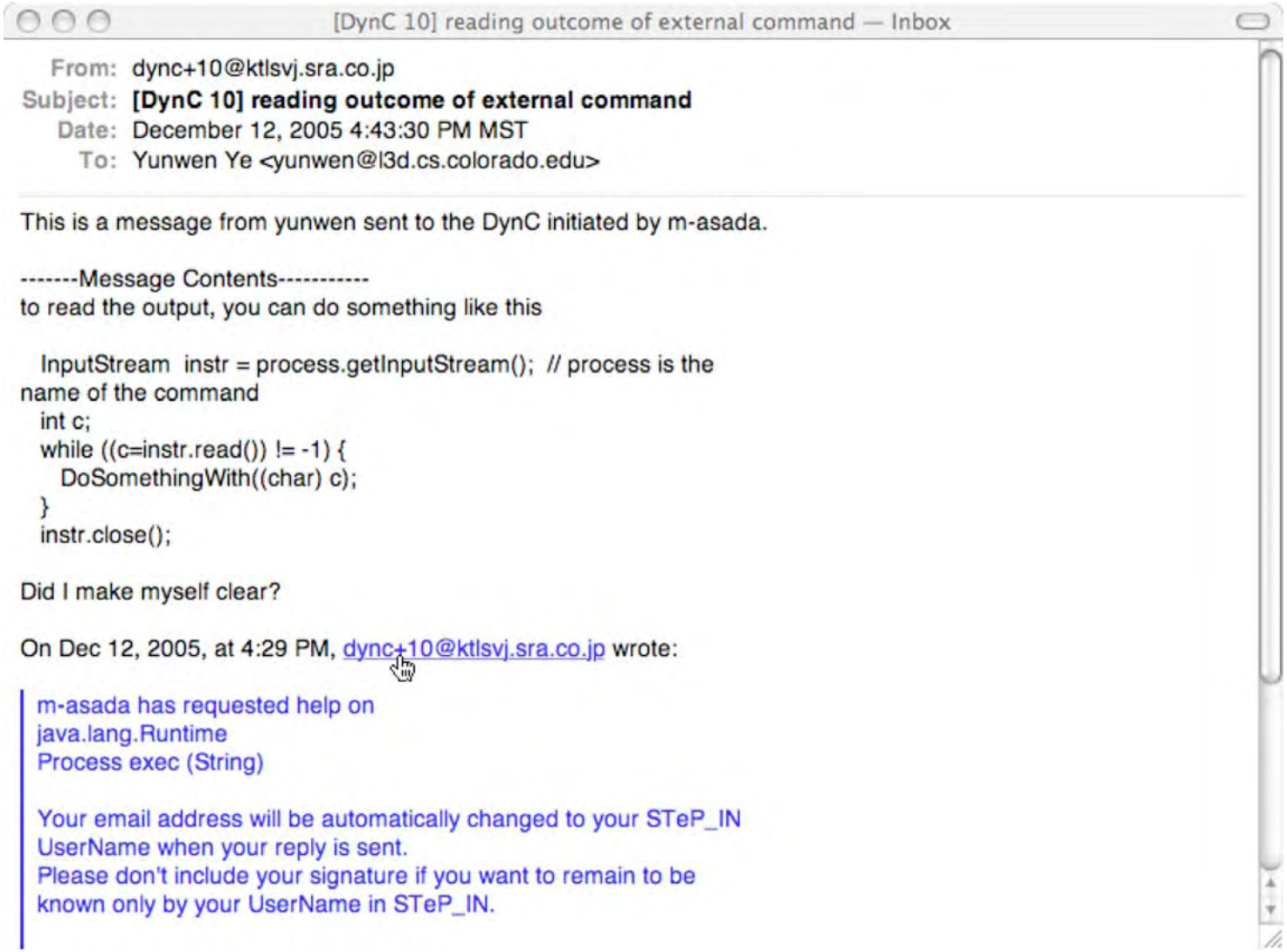
Ask the experts



Mail sent to selected experts



Reply from experts


[DynC 10] reading outcome of external command — Inbox

From: dync+10@ktlsvj.sra.co.jp
Subject: **[DynC 10] reading outcome of external command**
Date: December 12, 2005 4:43:30 PM MST
To: Yunwen Ye <yunwen@l3d.cs.colorado.edu>

This is a message from yunwen sent to the DynC initiated by m-asada.

-------Message Contents----------
to read the output, you can do something like this

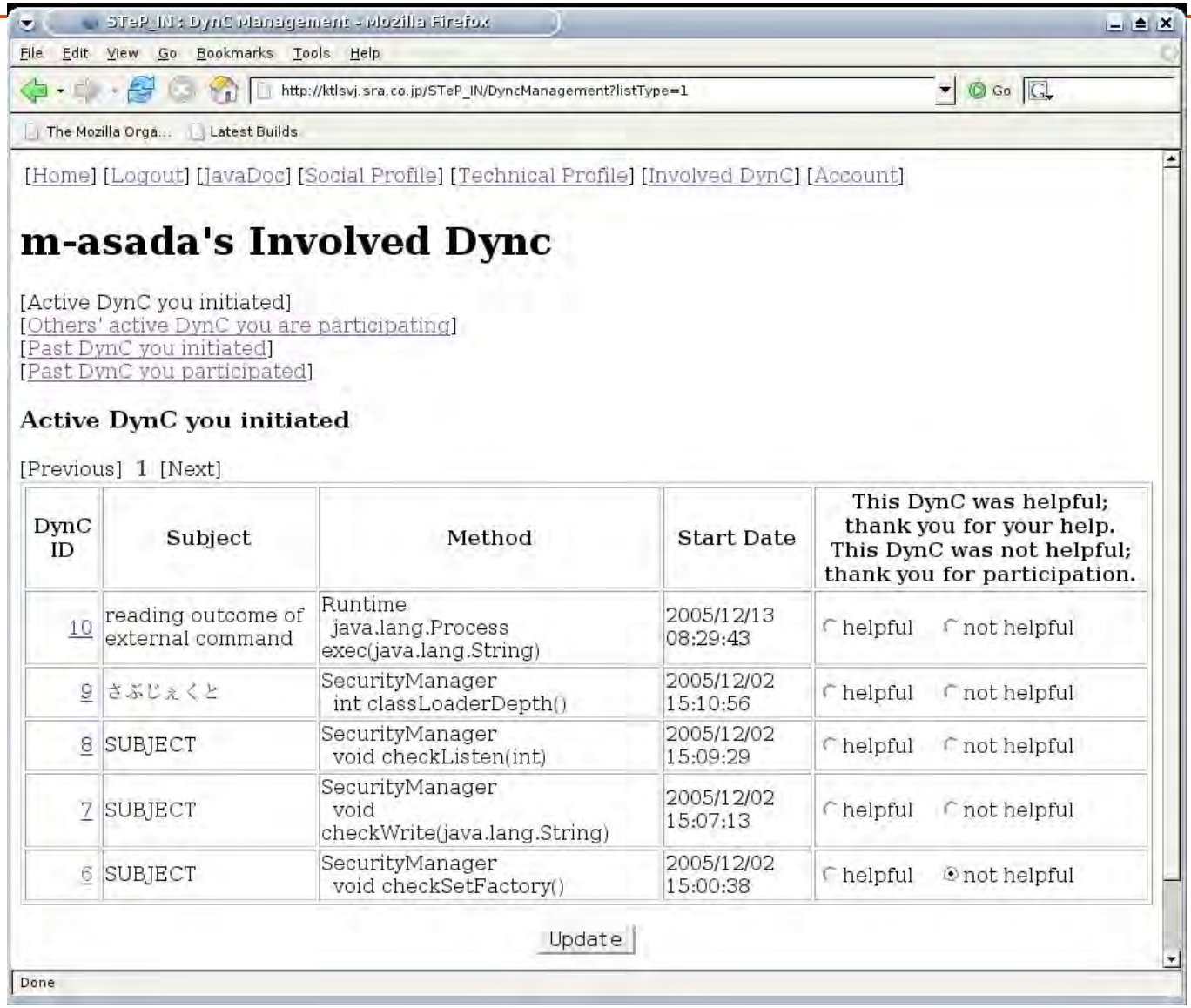
```
  InputStream  instr = process.getInputStream();  // process is the
name of the command
  int c;
  while ((c=instr.read()) != -1) {
    DoSomethingWith((char) c);
  }
  instr.close();
```

Did I make myself clear?

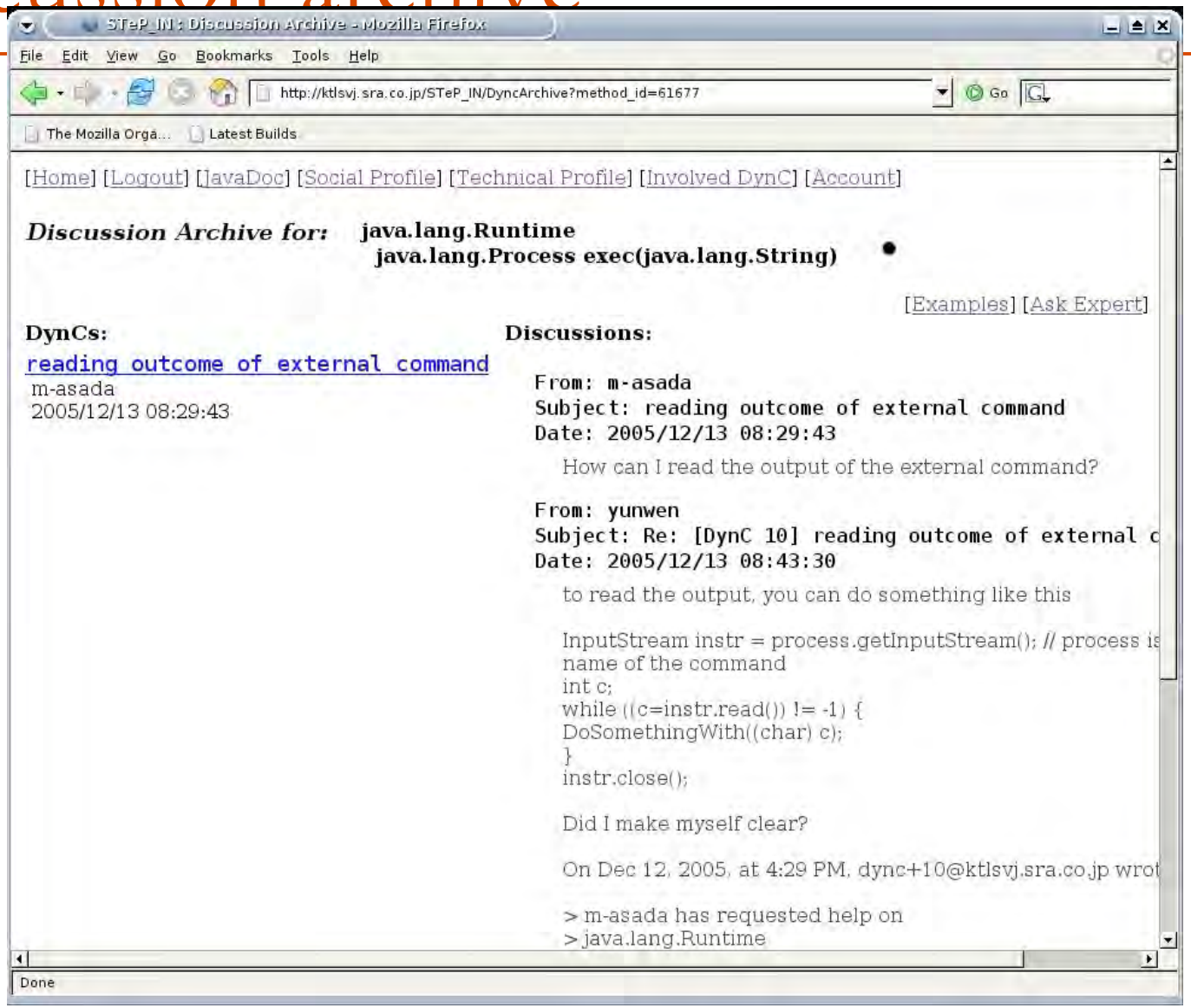
On Dec 12, 2005, at 4:29 PM, dync+10@ktlsvj.sra.co.jp wrote:

> m-asada has requested help on
> java.lang.Runtime
> Process exec (String)
>
> Your email address will be automatically changed to your STeP_IN
> UserName when your reply is sent.
> Please don't include your signature if you want to remain to be
> known only by your UserName in STeP_IN.

DynC evaluation



Discussion archive



DynC formation

- Expert identification
- Expert selection

Identifying experts

- Expert is a relative attribute
  - Only after a question is known, experts can be identified
- Creating user profiles by analyzing programs they have developed
  - Software developers who have used the method of interest are candidate experts
  - Links from methods to software developers

Expert selection

Level1: Confirmed expertise

Level2: Claimed expertise

Level3: Inferred expertise

Level4: Future expertise

Expert identification

Level1: confirmed expertise

vincent's Involved Dync

[Active DynC you initiated]
[Others' active DynC you are participating]
[Past DynC you initiated]
[Past DynC you participated]

Send email to DynC #6

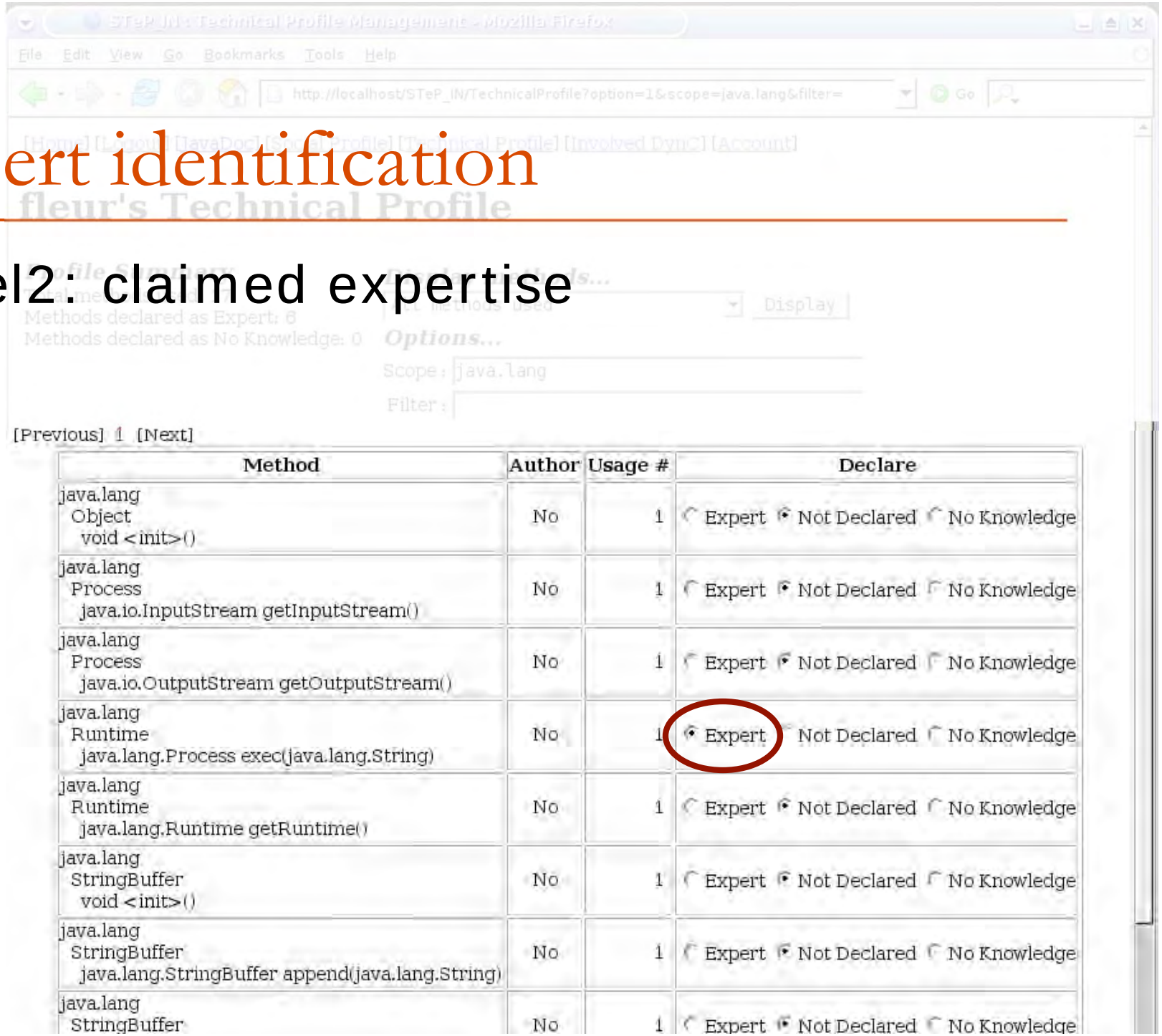
Past DynC you initiated

[Previous] 1 [Next]

DynC ID	Subject	Method	Start Date	End Date	Helpful/Not helpful
15	attributes of file after setReadOnly	File boolean setReadOnly()	2006/03/04 23:19:33	2006/03/04 23:25:06	helpful
8	help on exec	Runtime java.lang.Process exec(java.lang.String)	2006/03/02 16:02:08	2006/03/02 16:18:54	helpful
		Runtime			

Expert identification

Level2: claimed expertise

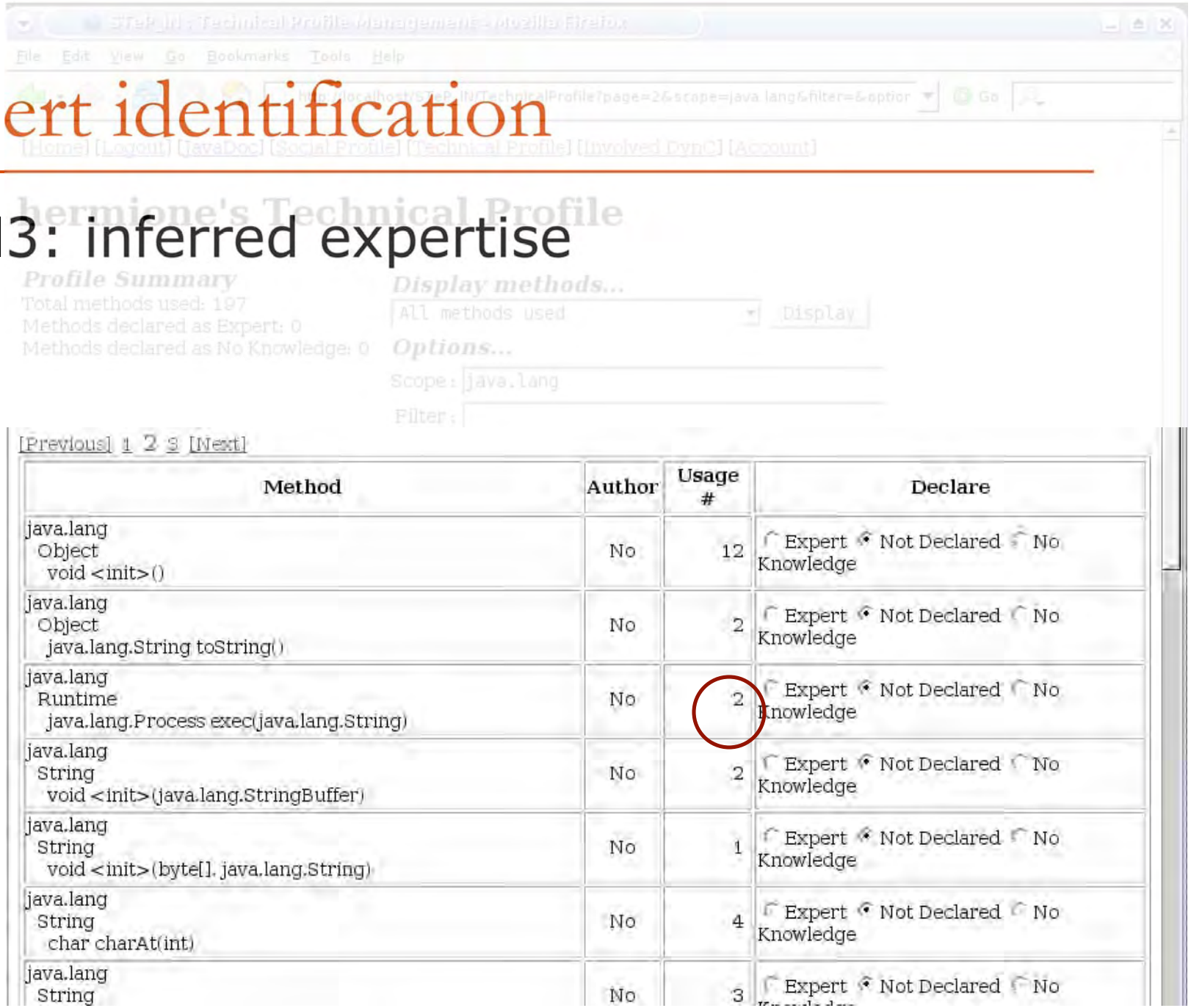


[Previous] 1 [Next]

Method	Author	Usage #	Declare
java.lang Object void <init>()	No	1	Expert Not Declared No Knowledge
java.lang Process java.io.InputStream getInputStream()	No	1	Expert Not Declared No Knowledge
java.lang Process java.io.OutputStream getOutputStream()	No	1	Expert Not Declared No Knowledge
java.lang Runtime java.lang.Process exec(java.lang.String)	No	1	Expert Not Declared No Knowledge
java.lang Runtime java.lang.Runtime getRuntime()	No	1	Expert Not Declared No Knowledge
java.lang StringBuffer void <init>()	No	1	Expert Not Declared No Knowledge
java.lang StringBuffer java.lang.StringBuffer append(java.lang.String)	No	1	Expert Not Declared No Knowledge
java.lang StringBuffer	No	1	Expert Not Declared No Knowledge

Expert identification

Level3: inferred expertise



[Previous] 1 2 3 [Next]

Method	Author	Usage #	Declare
java.lang Object void <init>()	No	12	Expert Not Declared No Knowledge
java.lang Object java.lang.String toString()	No	2	Expert Not Declared No Knowledge
java.lang Runtime java.lang.Process exec(java.lang.String)	No	2	Expert Not Declared No Knowledge
java.lang String void <init>(java.lang.StringBuffer)	No	2	Expert Not Declared No Knowledge
java.lang String void <init>(byte[], java.lang.String)	No	1	Expert Not Declared No Knowledge
java.lang String char charAt(int)	No	4	Expert Not Declared No Knowledge
java.lang String	No	3	Expert Not Declared No

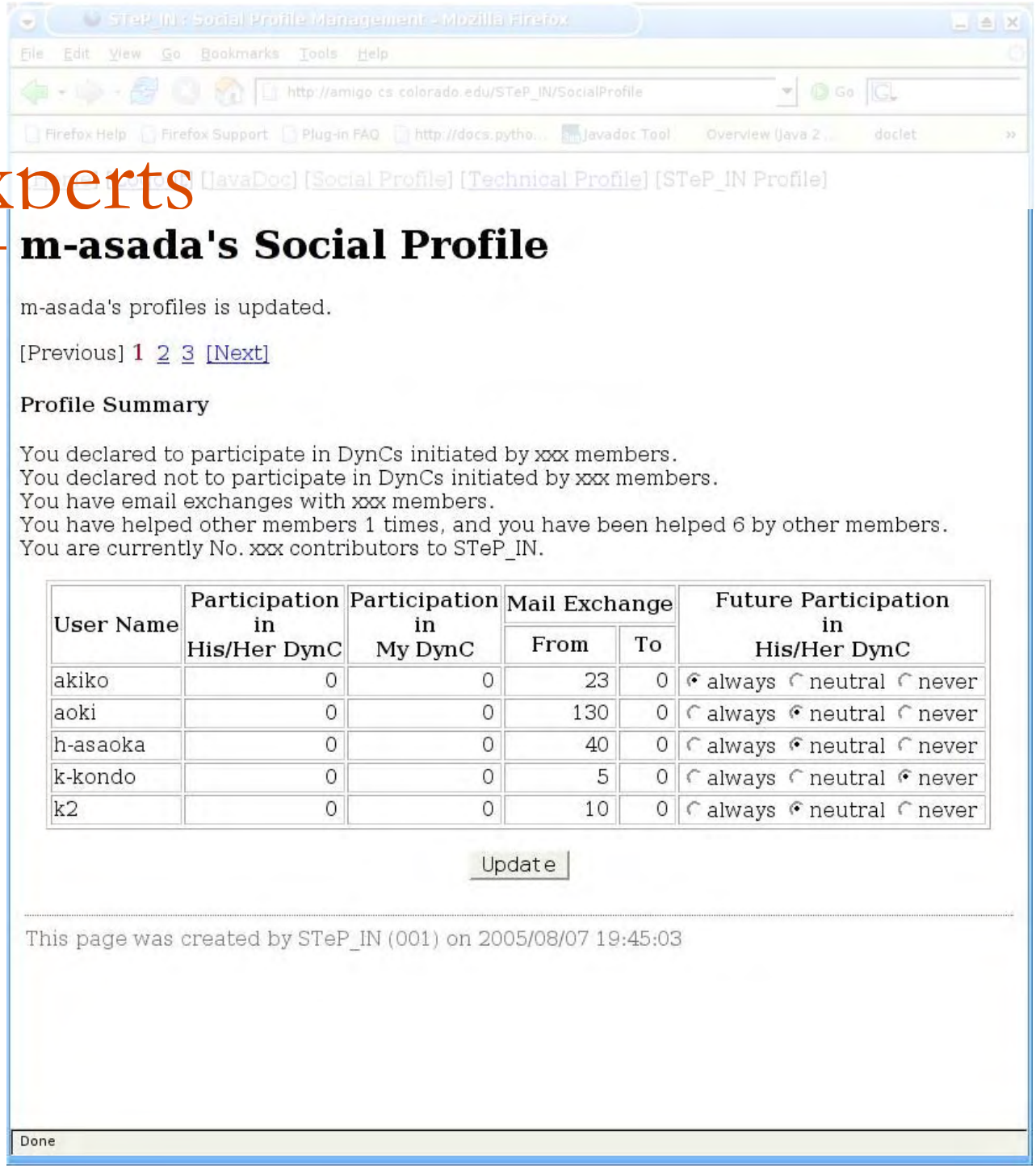
Selecting experts

- From expert candidates, select those who are most likely to help
  - Those I have helped recently
  - Those I have interacted through emails

Selecting experts

4 relations

help<A,B,t>
friend<A,B>
exclude<A,B>
email<A,B>



Selecting experts

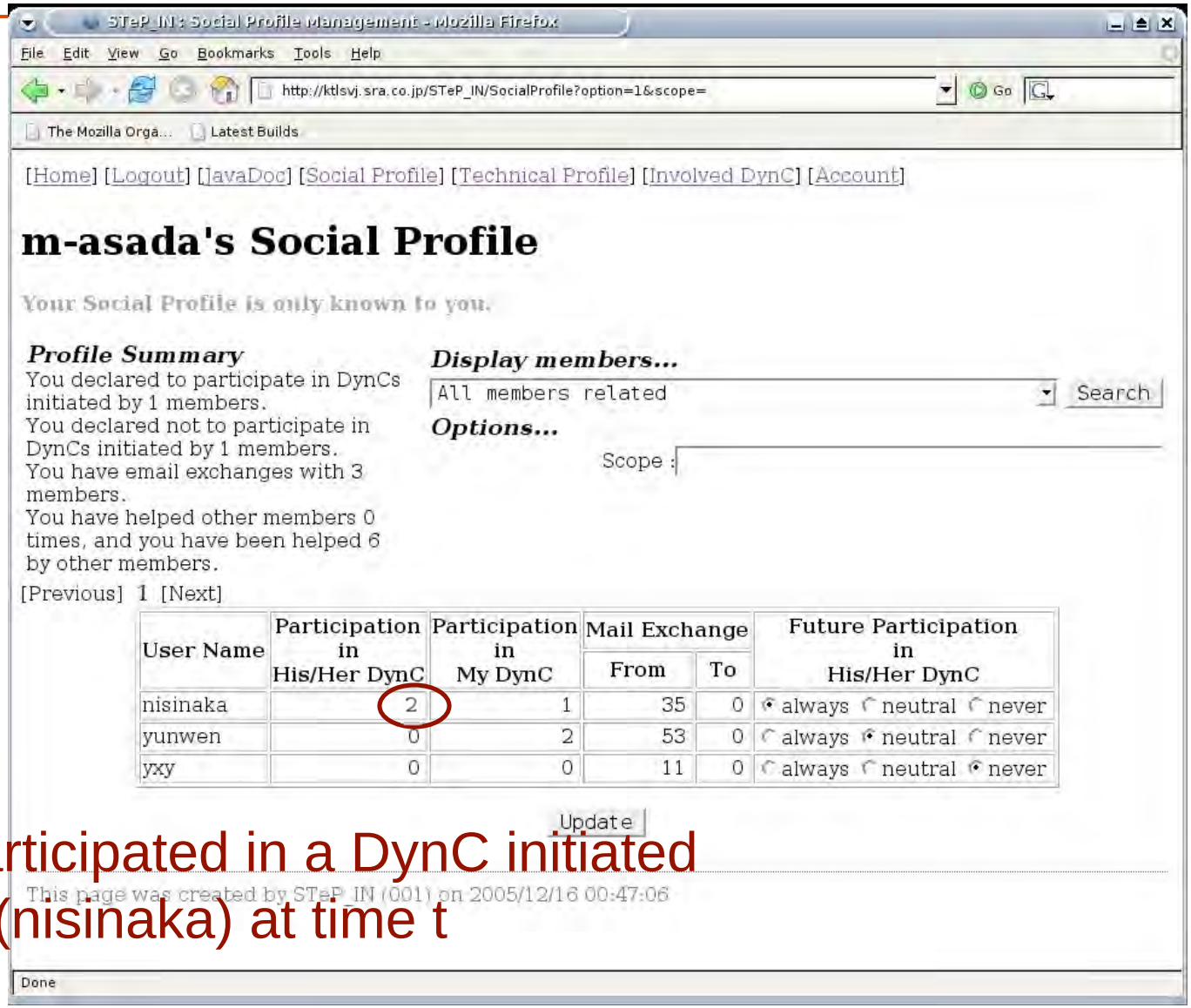
4 relations

help<A,B,t>

friend<A,B>

exclude<A,B>

email<A,B>



A (m-asada) participated in a DynC initiated by B (nisinaka) at time t

Selecting experts

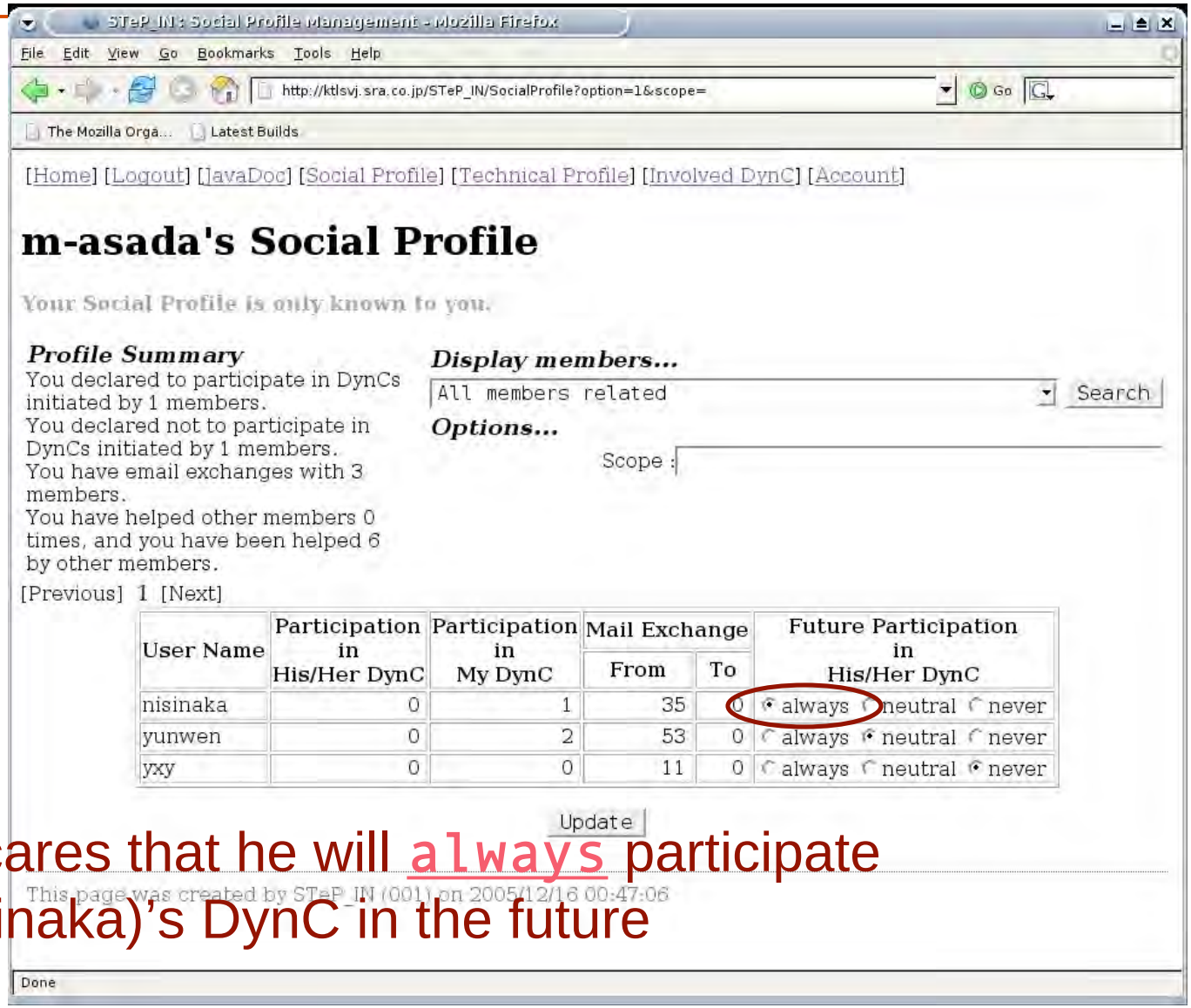
4 relations

help<A,B,t>

friend<A,B>

exclude<A,B>

email<A,B>

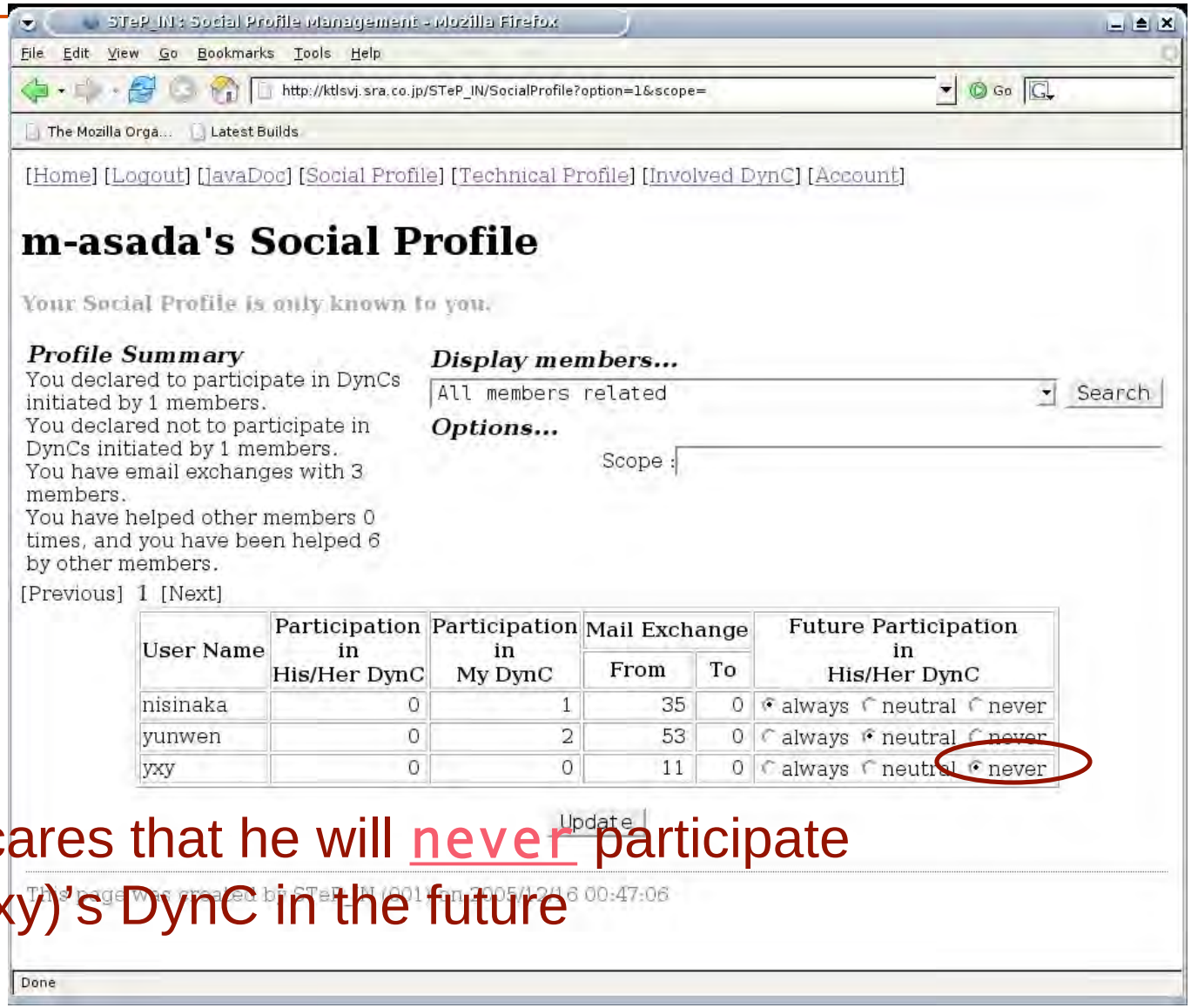


A (m-asada) delcares that he will always participate in B (nisinaka)'s DynC in the future

Selecting experts

4 relations

help<A,B,t>
friend<A,B>
exclude<A,B>
email<A,B>

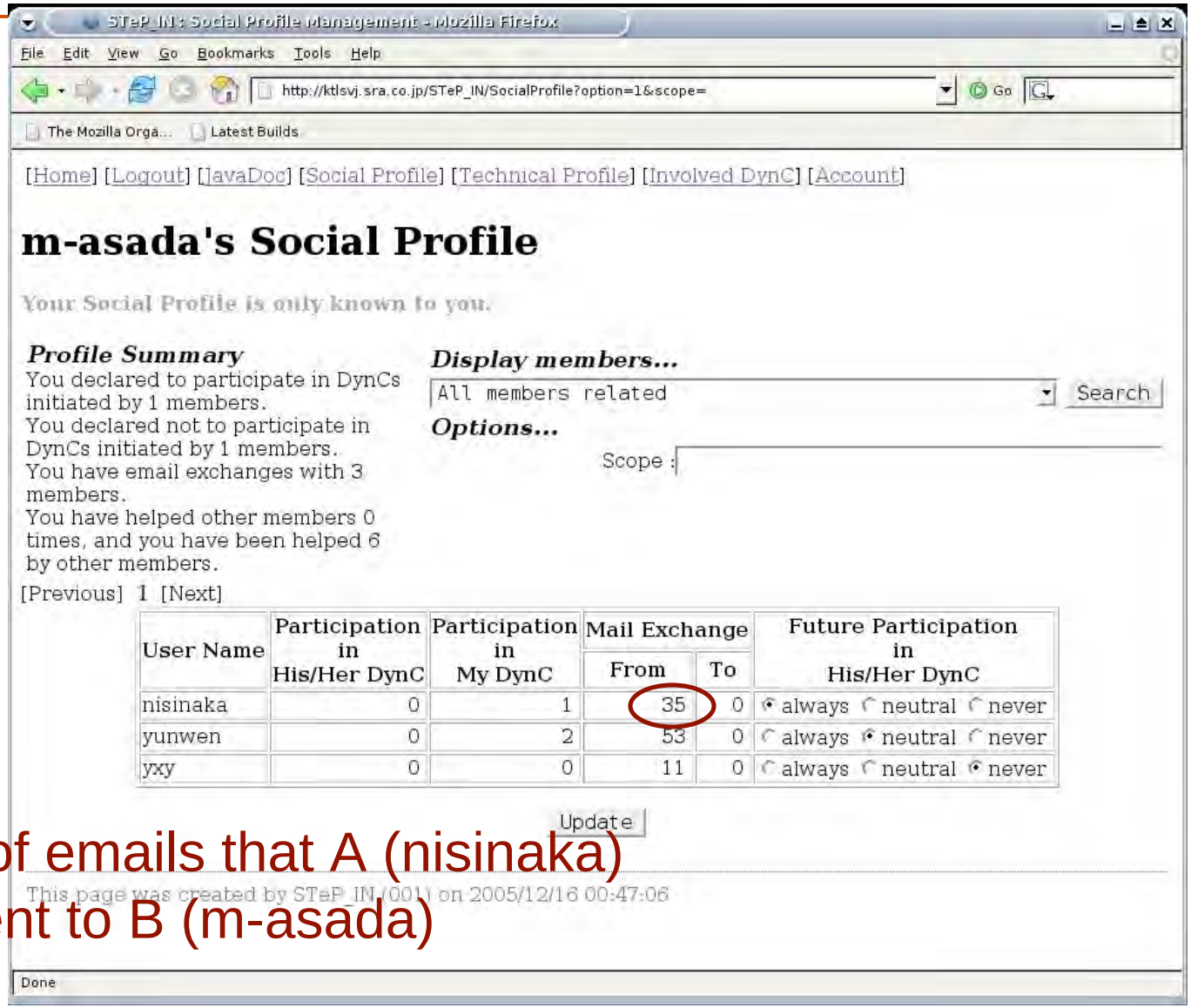


A (m-asada) delcares that he will never participate inB (yxy)'s DynC in the future

Selecting experts

4 relations

help<A,B,t>
friend<A,B>
exclude<A,B>
email<A,B>



The number of emails that A (nisinaka) has sent to B (m-asada)

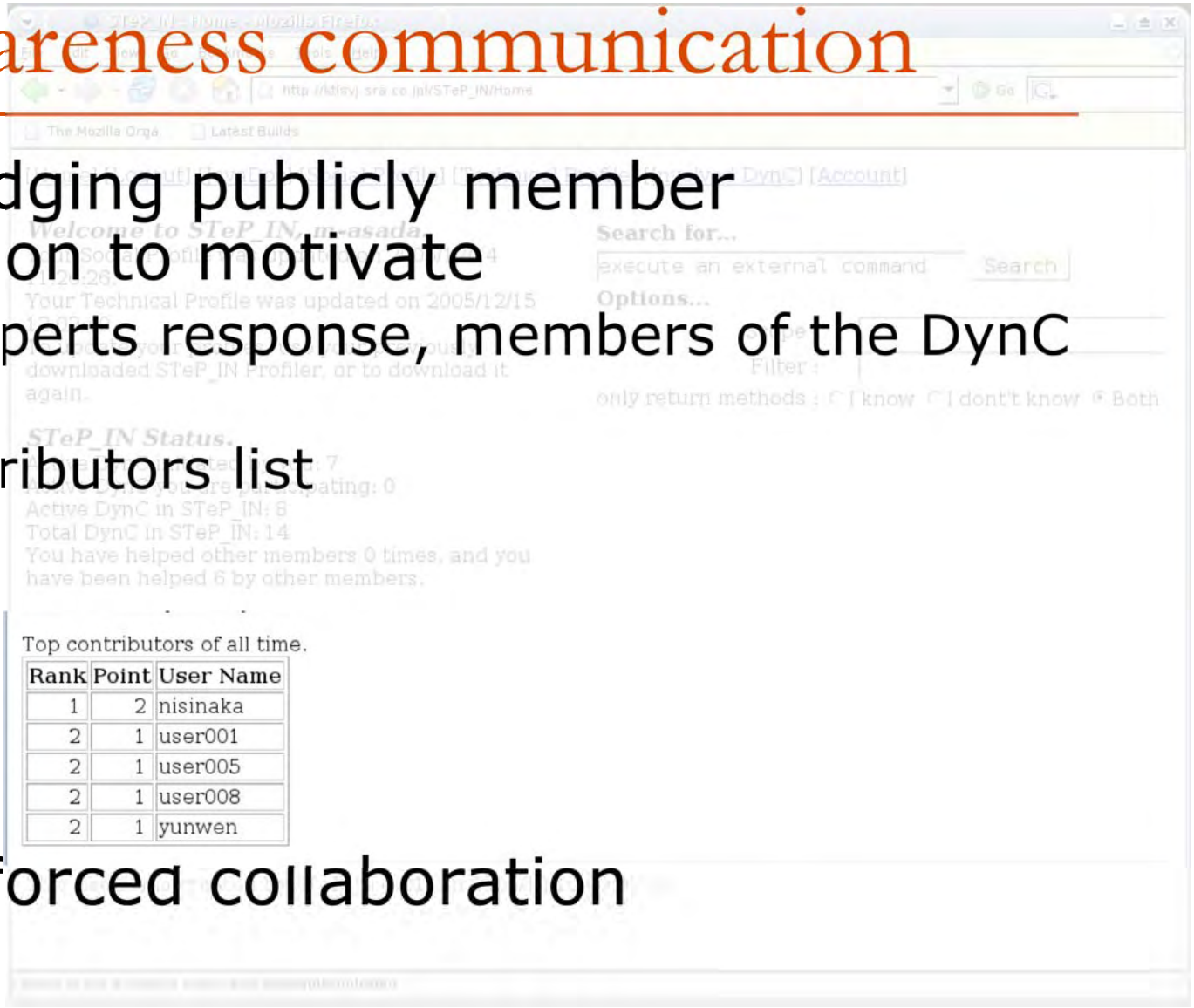
Selecting experts based on 4 relations

For each person X in identified expert lists

1. If exclude<X, A>, X is removed from the list
   because X declared he will never participate in A's DynC
2. If friend<X, A>, X is selected
   because X declared he will always participated in A's DynC
3. If |help<A, X, t>| > |help<X, A, t>|, X is selected
   because A has helped X more times than X did A
4. If help<A, X, t> and t is more recent than help<X, A, t>, X is selected
   because A has recently helped X
5. Selecting from remaining experts on the order of email<X, A>
   because X should have known A well if X has send many emails to A
6. Selecting from remaining experts those who got most help by other members

Social awareness communication

- Acknowledging publicly member participation to motivate
  - When experts response, members of the DynC know
  - Top contributors list



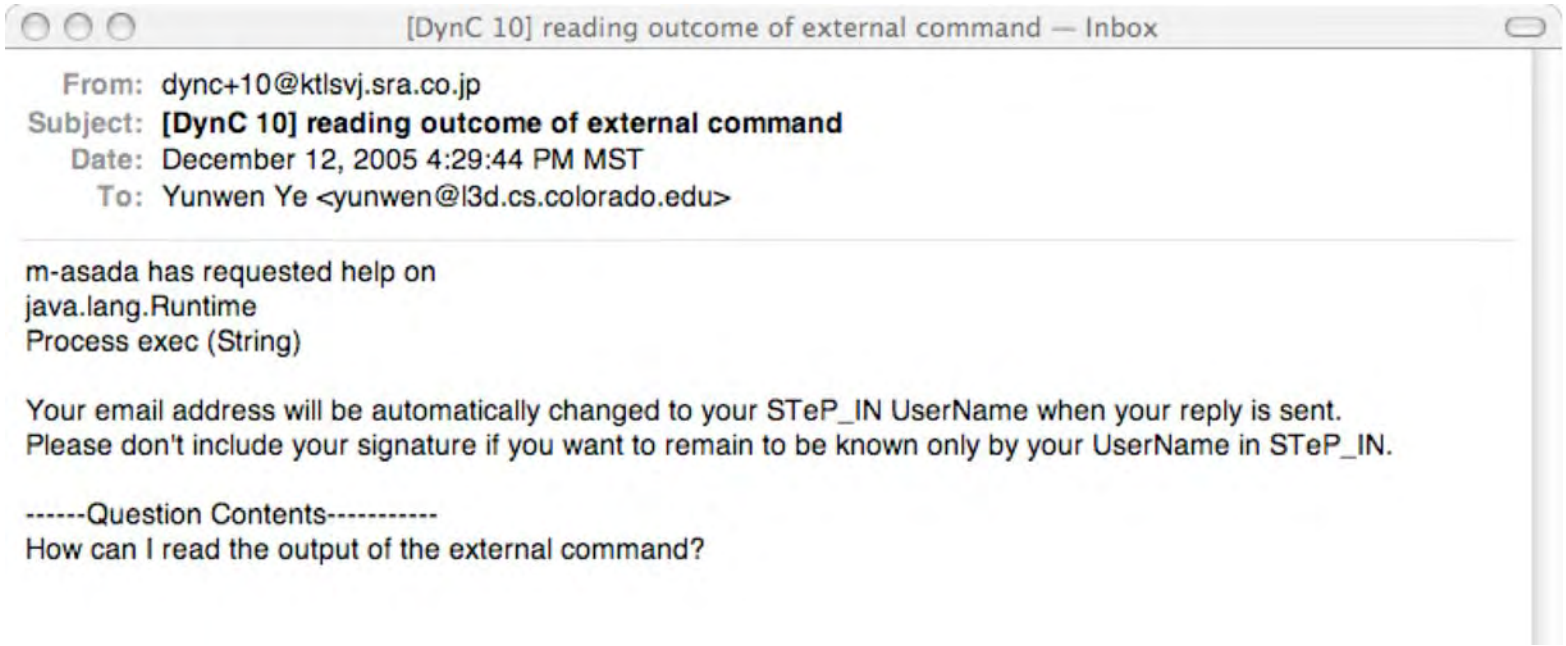
Top contributors of all time.

Rank	Point	User Name
1	2	nisinaka
2	1	user001
2	1	user005
2	1	user008
2	1	yunwen

- Avoiding forced collaboration

Avoiding forced collaboration

- When request for help is sent to experts, recipients are hidden from requesters and other experts


[DynC 10] reading outcome of external command — Inbox

From: dync+10@ktlsvj.sra.co.jp
Subject: **[DynC 10] reading outcome of external command**
Date: December 12, 2005 4:29:44 PM MST
To: Yunwen Ye <yunwen@l3d.cs.colorado.edu>

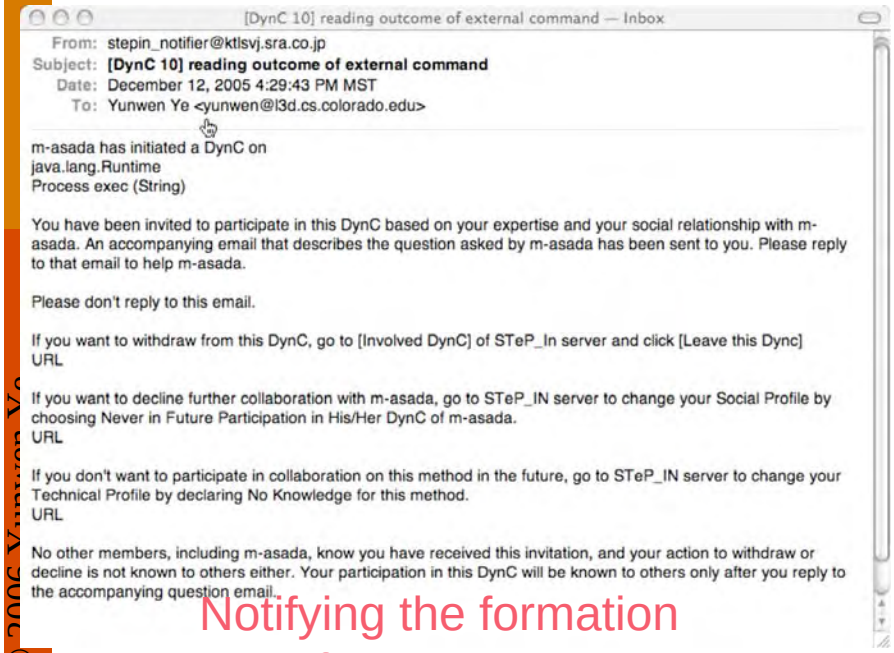
m-asada has requested help on
java.lang.Runtime
Process exec (String)

Your email address will be automatically changed to your STeP_IN UserName when your reply is sent.
Please don't include your signature if you want to remain to be known only by your UserName in STeP_IN.

------Question Contents----------
How can I read the output of the external command?

Avoiding forced collaboration

- Selected experts can quietly withdraw from the DynC without anyone noticing


[DynC 10] reading outcome of external command — Inbox

From: stepin_notifier@ktlsvj.sra.co.jp
Subject: **[DynC 10] reading outcome of external command**
Date: December 12, 2005 4:29:43 PM MST
To: Yunwen Ye <yunwen@l3d.cs.colorado.edu>

m-asada has initiated a DynC on
java.lang.Runtime
Process exec (String)

You have been invited to participate in this DynC based on your expertise and your social relationship with m-asada. An accompanying email that describes the question asked by m-asada has been sent to you. Please reply to that email to help m-asada.

Please don't reply to this email.

If you want to withdraw from this DynC, go to [Involved DynC] of STeP_In server and click [Leave this Dync]
URL

If you want to decline further collaboration with m-asada, go to STeP_IN server to change your Social Profile by choosing Never in Future Participation in His/Her DynC of m-asada.
URL

If you don't want to participate in collaboration on this method in the future, go to STeP_IN server to change your Technical Profile by declaring No Knowledge for this method.
URL

No other members, including m-asada, know you have received this invitation, and your action to withdraw or decline is not known to others either. Your participation in this DynC will be known to others only after you reply to the accompanying question email.

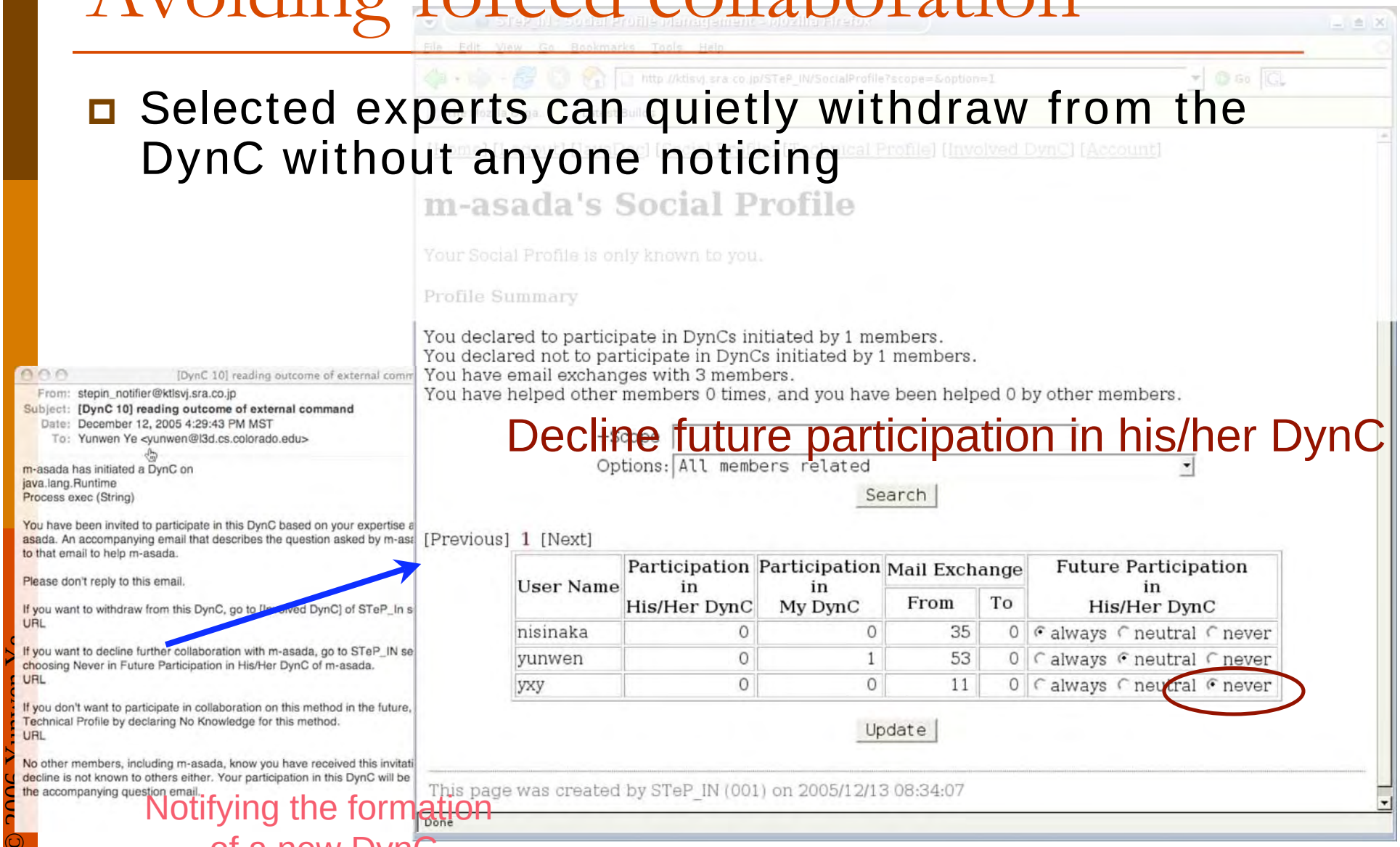
Notifying the formation of a new DynC

Avoiding forced collaboration

- Selected experts can quietly withdraw from the DynC without anyone noticing

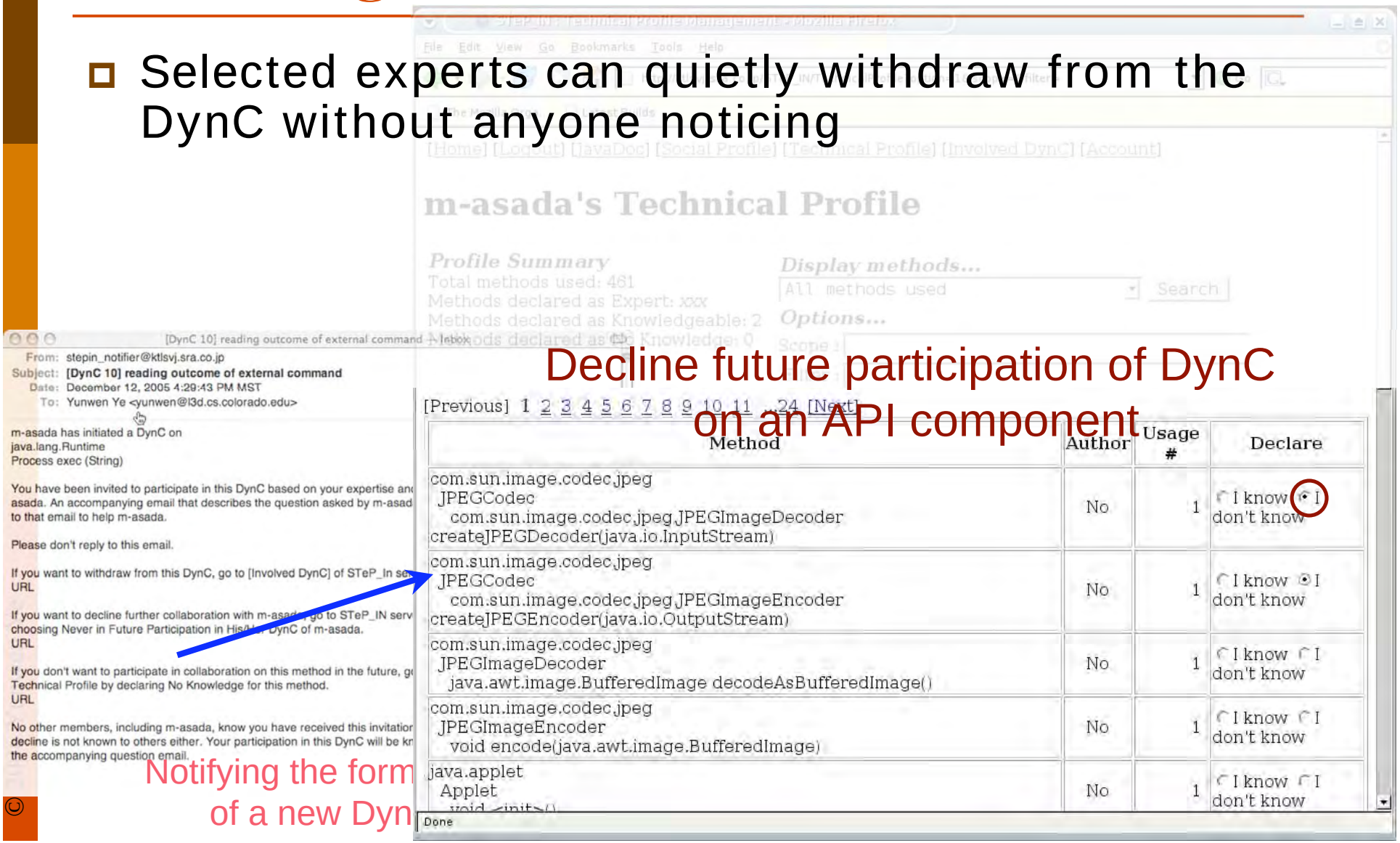
Avoiding forced collaboration

- Selected experts can quietly withdraw from the DynC without anyone noticing



Avoiding forced collaboration

- Selected experts can quietly withdraw from the DynC without anyone noticing



Social awareness communication

- Sending emails to ask the experts
  - Providing excuse space
    - A dose not know who are the recipients
  - Publicly acknowledgement
    - Any experts who answered the questions are made known
  - Easy exit
    - “Don’t bother me anymore about this problem”
    - “I don’t want to have more request emails from A”—establishing *exclude*<*X, A*>

Summary

- Better understanding of cognitive difficulties of component reuse
  - Unknown components
  - Low reuse utility
- A new type of component repository systems
  - Active component repository systems
- Integrating technology support with social support