ISSN 2079-3316 ПРОГРАММНЫЕ СИСТЕМЫ: ТЕОРИЯ И ПРИЛОЖЕНИЯ № 4(8), 2011, с. 115–126

УДК 004.651.53:004.827

С. В. Знаменский

Архитектура коллаборативно-изменяемой иерархической структуры

Аннотация. Прорабатывается новый подход к коллаборативной навигации по большой неоднозначно структурируемой коллекции разнородных ресурсов, осуществляемой большим числом пользователей. Важным отличием от обычного краудсорсинга является полная децентрализация управления.

Подход базируется на выделении и объединении усилий пользователей-единомышленников, направленных на классификацию коллекции. Нестандартно мыслящий пользователь потратит адекватные усилия на формирование индивидуальной таксономии, точной в зоне его непосредственных интересов. Чем активнее пользователь работает в интерфейсе и чем больше у него активных единомышленников, тем быстрее и совершеннее таксономия подстроится под его представления.

Среди пользователей могут быть роботы, анализирующие ресурсы.

Описываются общая архитектура будущей системы и отдельные алгоритмы.

Ключевые слова и фразы: коллаборативное ПО, архитектура ИС, каталог ресурсов, таксономия, классификация.

Введение

Совершенствование любого пользовательского интерфейса связано с оптимизацией навигации. Поскольку основой практически любой пользовательской навигации являются иерархические структуры, то построение динамических таксономий является базой любых пользовательских интерфейсов. Для актуальных ситуаций больших коллекций ресурсов задача построения качественной таксономии практически неразрешима усилиями небольшой рабочей группы. Поэтому всё большее внимание исследователей приобретают (см. подробнее в [1–3]) широко распространяющиеся облака тегов и родственные

Работа проводилась при финансовой поддержке Министерства образования и науки Российской Федерации и при поддержке РФФИ, грант № 09-07-00407.

механизмы включения неорганизованных пользователей в работу по структурированию ресурсов.

Более глубокий, чем облака тегов, подход кратко намечен в [4]. Суть его в индивидуальной классифицирующей деятельности, в ходе которой каждый участник получает персональную классификацию, а все вместе — компромиссное решение. При этом наличие активных единомышленников многократно ускоряет процесс.

Выполнимость требования прозрачной однозначности идентификации контекста становится проблематичной для особо больших таксономий, где к одному родовому понятию могут соотноситься десятки семейств видов, либо десятки видов могут наблюдаться в одном подсемействе. По-видимому, единственно разумным путём преодоления этой сложности является введение нового уровня классификации либо расширение на верхнем уровне (введение нового рода, семейства, etc.) Проблема поиска правильного основания для классификации может оказаться крайне трудно разрешимой для больших разнородных коллекций. Поэтому с любым контекстом должна легко связываться вики-страничка или иной удобный ресурс для обсуждения, и должна присутствовать возможность отложить чёткое решение этого вопроса, допустив перекрытие областей дочерних контекстов (здесь дочерним назван контекст, получающийся выбором пункта меню).

Строго говоря, это означает, что таксономия не получится вполне корректной в том смысле, что некоторые ветки иерархического дерева наложатся друг на друга. Это легко исправить, отнеся пересечение к первому по алфавиту тегу, но такое исправление повысит риск не найти в каталоге имеющийся ресурс. Если не получится выработать единственно правильную таксономию, то получится единая, наиболее устраивающая большинство участников процесса, плюс индивидуальные версии участников и возможность продолжить работу. Бывает полезно получить одновременно несколько общеупотребительных вариантов, оптимизированных для разных ситуаций использования. Например, для некоторых устройств или режимов работы важно существенно ограничить размер меню, что приведёт к неоптимальному в менее стеснённых условиях варианту таксономии.

Несмотря на очевидную привлекательность и удобство пользования, эта идея ранее не прорабатывалась вероятно по очевидной причине расточительной ресурсозатратности в случае реализации стандартными средствами. Требуется непрерывно собирать, хранить и

совместно обрабатывать огромные объёмы не денормализуемой естественным образом информации о пользовательской активности и решать задачи неимоверной вычислительной сложности. Развитие технологий и удешевление вычислительных ресурсов открывает новую перспективу для улучшения пользовательского интерфейса.

Целью исследования является проработка общей архитектуры хорошо масштабируемой системы для совместного редактирования таксономии. В первом разделе намечены основы организация данных, во втором — основы организации обработки данных.

1. Логика пользовательского интерфейса, основанного на таксономии

Имеется пополняемая коллекция ресурсов

$$\mathbf{R} = \{r_1, r_2, \ldots, r_{|\mathbf{R}|}\}$$

и пополняемое множество классифицирующих признаков или тегов (разделов, подразделов коллекции, родов, видов, семейств, атрибутов ресурсов)

$$\mathbf{T} = \{t_1, t_2, \ldots, t_{|\mathbf{T}|}\}.$$

Выбирая разделы, пункты меню, кнопки и т.д. пользователь получает *контекст* $c = (I, X) \subset \mathbf{C}$ который в конкретный момент характеризуется множеством включающих (выбранных) тегов $I \subset \mathbf{T}$ и множеством при этом исключённых (недоступных для дальнейшего уточнения выбора) тегов $X \subset \mathbf{T} \setminus I$. Множество контекстов частично упорядочено отношением узости контекста: $c_1 \leqslant c_2$ означает $I_1 \supset I_2$ и $X_1 \supset X_2$.

Таким образом, контекст определяется не последовательностью действий пользователя, а их логикой. Это обеспечивает сохранение всей работы по классификации при любых перестройках таксономии.

Контекст c определяет множество $R_c(c) \subset \mathbf{R}$ ресурсов, которые доступны дальнейшим уточнением выбора. Если $c_1 \leqslant c_2$, то $R_c(c_1) \subset R_c(c_2)$. Множество $R_c(c)$ будем при этом называть областью контекста c, а его мощность — мощностью контекста. Множество выбранных тегов обычно показывается пользователю в виде текущих заголовков, выбранных пунктов контекстного меню или «хлебных крошек», множество невыбранных тегов обычно отображается невыбранными заголовками.

В выбранном контексте имеется подразделы и(или) меню, заголовки и пункты которых являются тегами, осуществляющими дальнейшую классификацию. Классифицирующие теги меню могут подразделяться на классификаторы — группы несовместимых тегов, осуществляющие классификацию по независимым признакам. Для простоты его составляющие и весь набор будем упорядочивать по алфавиту. Классификатор $k \subset \mathbf{T}$ пригоден для использования в контексте $c = (I, X)$, если $k \cup I \cup X = \emptyset$.

Список ещё не расклассифицированных объектов контекста упорядочивается по убыванию ценности (см. раздел 2.3) и обрезается. Все *списки* любых объектов, упоминаемые в этом разделе, подобным образом оформляются перед выдачей.

Выбранный тег может подразумевать ранее выбранные, как, например, латинское название семейства обычно относится к уникальному роду. Это делает постоянную индикацию подразумеваемых тегов излишней. Чтобы не загромождать рабочий экран, лишние теги можно спрятать в иконку, наведение на которую покажет, а клик развернёт спрятанные теги. С точки зрения логики не важно, как конкретно будет оформлена навигация.

Однако для недвусмысленности интерпретации действий пользователя важно, чтобы в любом варианте интерфейса работы со *списком* релевантных объектов в избранном контексте пользователь в каждый момент видел ясное и полное описание контекста $C \in \mathbf{C}$, в котором он действует.

1.1. Интерфейс модификации таксономии

Чтобы индивидуальность личности могла свободно проявиться, желание пользователя поправить классификацию должно легко и полно реализовываться в интерфейсе. Входящие в коллекцию ресурсы могут помечаться на соответствие тем или иным тегам. Это означает, что пользователь должен иметь следующие возможности:

(1) закрепить ресурс в данном контексте (нажав кнопку или пометив чекбокс);
(2) удалить ресурс в конец списка с пометкой низкой оценки (в дальнейшем он может исчезнуть);
(3) переместить ресурс в иной контекст (например, захватив мышкой и положив на соответствующий тег);
(4) добавить тег в контекстный классификатор;
(5) убрать тег из контекстного классификатора;

(6) изменить формулировку тега в контекстном классификаторе;
(7) изменить классификатор;
(8) сменить комплект классификаторов для контекста.

Важно, чтобы интерфейс обеспечил однозначность логической интерпретации этих действий пользователем и возможность отмены ошибочных действий. Не менее важно обеспечить подсказки, упрощающие выполнение перечисленных действий:

(1) Зрительно выделять ресурсы, значимость которых в данном контексте единодушно подтверждена пользователями, а также спорные, негативно оцененные ресурсы и показать убедительность[1] оценок.
(2) При наведении мышкой на ресурс должен показываться *список* тегов контекста, спорно ассоциируемых с ресурсом, в порядке убывания неоднозначности.
(3) Зрительно выделять ресурсы, оценённые самим пользователем.
(4) Интерфейс добавления тега в классификатор должен предлагать *список* вариантов.
(5) При наведении мышкой на спорный тег классификатора должна показываться оценка спорности и быть доступен механизм удаления тега.
(6) При наведении мышкой на тег классификатора должны показываться имеющиеся варианты переименования, а при клике возникать меню переименования/удаления с селектом и полем ввода.
(7) При наличии альтернативных вариантов замены классификатора контекста или комплектов классификаторов должна быть возможность их просмотреть и сопоставить.
(8) Очень полезна индикация количества ресурсов в каждой градации классификатора (минимального и дублирующегося), а для каждого из ресурсов — индикация полного списка назначенных ему тегов.

Любое действие персоны может рассматриваться как оценка таким же (и противоположным) предшествовавшим действиям его самого или других пользователей. Наличие полной системы оценок

[1] убедительность оценки — это доля положительно оценивших среди оценивавших, расчитанная с учётом авторитетности

призвано обеспечить выделение контекстно более авторитетных пользователей, придание их действиям большего приоритета, автоматическое игнорирование непрофессиональных мнений и обеспечение возможностей независимой отработки альтернативных подходов к классификации.

2. Базовые структуры данных

Для уменьшения дублирования в представлении многих слабо различающихся ревизий объёмных данных удобно разбивать их на короткие записи вида «ключ-значение» об изменениях. Ключи таких записей конкатенируются из логического ключа и строки, идентифицирующей версию и время записи.

Логические ключи записей о присвоении или изменении значений формируются конкатенацией идентификатора функции или отношения и идентификаторов аргументов. Идентификатор любой сущности формируется из символа, идентифицирующего тип сущности и случайной строки, длина которой зависит от типа.

Представляется разумным накапливать и использовать информацию следующих видов:

2.1. Идентификаторы базовых сущностей

Пользователи. Учитывая количество жителей Земли, экономно идентифицировать пользователя $u \in U$ разумно 4-байтной строкой, которая в дальнейшем будет обозначаться той же буквой u. Вся информация, относящаяся к пользователю u, будет храниться в записях, ключи которых начинаются на букву «`u`», продолженную строкой u. Например, логический ключ `uUSERn` будет указывать на полное имя пользователя с идентификатором «`USER`».

Ресурсы. Ресурс — это объект, который должен иметь уникальный идентификатор $r \in R$, размер которого заведомо можно ограничить 6 байтами (десятки тысяч ресурсов на каждого участника). Вся информация о ресурсе будет храниться в записях, ключи которых начинаются на букву «`r`», продолженную строкой r. Например, логический ключ `rRESOURu` будет указывать на `url` ресурса с идентификатором «`RESOUR`».

Теги. Поскольку каждый тег есть не что иное как ключевая фраза, то есть строка символов, то будем считать, что теги идентифицируются этими строками. Для экономии вводятся более компактные четырёхбайтные идентификаторы (предельное количество таких идентификаторов превосходит словарный запас любого существующего словаря). Например, ключ `tTEG1k` будет указывать на ключевую фразу, имеющую идентификатор «`TEG1`», пусть это будет «`Пример тега`».

Контексты. Мощность множества возможных контекстов легко выходит за рамки практических возможностей непосредственной адресации. Поэтому приходится учитывать лишь реально использованные контексты, идентифицируя их начинающейся на «`c`» случайной строкой из 6 байт (сотни на каждого жителя земли). Например, ключ `cCONTEXl` будет указывать на пару алфавитно упорядоченных списков идентификаторов тегов контекста.

Классификаторы. Классификатор — это упорядоченное множество тегов. Естественно ожидать, что количество различных классификаторов будет расти не быстрее, чем количество различных тегов. Например, ключ `kCLS1l` будет указывать на список ключей тегов для классификатора с идентификатором «`CLS1`».

Хотя информация о пользователях и метаинформация о ресурсах может уточняться, остальные сущности остаются неизменными, а изменения производятся через замену идентификатора.

2.2. Пользовательский ввод

Применяя описанный выше интерфейс, пользователь $u \in U$ придаёт значения следующим функциям:

(1) Позиционирование ресурса в начале списка: $P_{ucr}(u, c, r) = 1$.
(2) Удаление ресурса в конец списка: $P_{ucr}(u, c, r) = 0.25$.
(3) Перемещение ресурса P в новый контекст c':

$$P_{ucr}(u, c', r) = P_{ucr}(u, c, r); \quad P_{ucr}(u, c, r) = 0.$$

(4) Добавление классифицирующего тега в контексте:

$$F_{uct}(u, c, t) = 1.$$

(5) Удаление классифицирующего тега из контекста: $F_{uct}(u, c, t) = 0$.
(6) Замена классифицирующего тега t на t' в контексте:

$$F_{uct}(u, c, t) = -1; \quad F_{uct}(u, c, t') = +1.$$

(7) Запись нового комплекта классификаторов v.
(8) Выбор комплекта v классификаторов $V_{uc}(u, c) = v$.

2.3. Вторичные данные и некоторые алгоритмы их вычисления

Вторичные данные уточняются по мере изменения основных данных. Быстрее всего должны изменяться индексы.

Индексы необходимы для поиска идентификатора по частичной или полной информации. Идентификаторы индексных записей начинаются на «`i`», продолжаются буквой искомого, описанием строки поиска и самой строкой поиска. В частности, ключ `itkПример тега` должен иметь значение «`TEG1`», а ключ «`iklTEG1`» даст идентификатор классификатора, состоящего из единственного тега «`Пример тега`».

Список ближайших[2] *более узких контекстов* $C' \in \mathbf{C}$ по отношению к контексту C.

Список ближайших более широких контекстов $C' \in \mathbf{C}$ по отношению к контексту C.

Различные сводные оценки могут вычисляться время от времени и уточняться по изменяющимся правилам.

Авторитетность — это коэффициент, с которым учитываются мнения пользователя. Пользователь по умолчанию имеет авторитетность $E_u(u) = 1$, но эта оценка может быть изменена записью с ключом вида `euUSER`. Причинами могут быть как воздействие администраторов, так и статистика повторения его действий либо противодействия им со стороны других пользователей.

Популярность контекста — это количество пользователей, обратившихся к контексту.

Контекстная ценность ресурса $E_{cr}(c, r) = 0.5$ для только что добавленного пользователем ресурса, но может измениться в границах $0 \leqslant E_{cr}(c, r) \leqslant 1$. Для этого все пользовательские оценки $P_{ucr}(u, c, r)$ ресурса в контексте усредняются с весом авторитетности пользователя и записываются с ключом вида `ecrCONTEXRESOUR`.

Ценность ресурса $E_r(r)$ умолчательно считается равной 0.5, но периодически уточняется на основе усреднения контекстной ценности ресурса с весом популярности контекстов. Результаты усреднения записываются с ключом вида `erRESOUR`.

[2] В том смысле, что между ними нет третьего контекста

Явное несогласие $Z_{ucc}(u, c_1, c_2) = 1$ пользователя u с *вложенностью областей контекстов* $\exists r : r \in R_c(c_1) \wedge \neg(r \in R_c(c_2))$, проявляющееся в существовании $c_3 \geqslant c_2$, $c_4 \leqslant c_1$ и r, для которых $P_{ucr}(u, c_3, r) = 0$, $P_{ucr}(u, c_4, r) = 1$. При отсутствии явного несогласия $Z_{ucc}(u, c_1, c_2) = 0$.

Разность $Z_{cc}(c_1, c_2) \in [0, 1]$ контекста c_1 и c_2 означает общее несогласие пользователей с тем, что каждый ресурс, лежащий в c_1, лежит и в c_2. Она равна 0 если $c_1 \leqslant c_2$, и 1, если все пользователи не согласны, и в общем случае вычисляется как взвешенное среднее от $Z_{ucc}(u, c_1, c_2)$ с весом E_u. Эта характеристика различий контекстов может быть использована при формировании подсказок. Они могут предлагать поместить ресурс в контекст, если другие поместили его в практически не больший. Или наоборот, предложить использовать в контексте принятый классификатор из близкого (практически не меньшего) контекста. Это позволит использовать результаты проработки одного варианта таксономии в другом варианте и повысит эффективность сотрудничества.

Персональный выбор классификатора в контексте $K_{uc}(u, c) = \{t : F_{uct}(u, c, t) = 1\}$.

Препочтительность классификатора в контексте для подсказок пользователям может сначала производиться по простой формуле:

$$E_{kc}(k, c) = \frac{|\{u : K_{uc}(uc) = k\}|}{2 + |\{u : \exists K_{uc}(uc)\}|} + \frac{|\{u : \exists c', K_{uc}(uc') = k\}|}{8 + 4 * |\{u : \exists c', \exists K_{uc}(uc')\}|} + \frac{3}{|k| + 3},$$

а в дальнейшем по более сложным, учитывающим выступание.

Организация подсказок, основывающихся на действиях пользователей, в наиболее близких ситуациях потребует заготовок ряда более сложных структур.

Полное рассмотрение структур данных и используемых алгоритмов далеко выходит за рамки настоящей статьи. Целью её ставилось дать чёткое представление о характере основных логических структур и архитектурных принципов реализации.

Среди них немало требующих учёта обновлений информации из заранее не локализуемых источников.

3. Управление обработкой данных

Требования к обработке данных противоречивы:

- интерфейс должен быть живым и мгновенно реагирующим на действия пользователя;

- он должен быть логически целостным, свободным от алогизмов;
- он должен содержать дружественные подсказки, формируемые на основе деятельности в близких контекстах похоже мыслящих в близких контекстах пользователей.

Близость контекстов определяется размером разности между ними, обновляемой по итогам активности пользователей, отыскать которые в объёмных данных непросто. Например, пользователь, отнеся ресурс к контексту, может войти в противоречие с другим, отнёсшим этот же ресурс к другому контексту и третьим, считающим эти контексты несовместимыми. Большие объёмы данных делают задачу формирования подсказок ресурсозатратной. Чтобы решать её без задержек, приходится отказаться не только от стандартной методологии, основанной на реляционной СУБД, но и от более изощрённых попыток сохранения свойств ACID, таких как доски объявлений [5].

Отказ от тотальной ежесекундной согласованности данных позволяет найти новый подход к обеспечению быстрого безупречно адекватного реагирования на действия пользователя. Фокус в том, что каждому пользователю важно, чтобы система быстро реагировала на его действия, но доставляет сомнительное удовольствие когда рабочий стол живёт своей жизнью. Поэтому пользователю удобно работать с застывшим состоянием системы, которое меняет он один, а обновления исследовать при новом входе в систему. Тогда обновления могут обрабатываться в фоновом режиме, параллельно и распределённо, и доводиться по мере готовности, и загружаться на компьютер пользователя в виде, позволяющем быстро и согласованно накладывать локальные изменения и предупреждать легко диагностируемые ошибки.

Сложно диагностируемые логические несоответствия будут выявляться по мере обработки и учитываться в следующих сессиях. Таким образом, процесс поддержки текущей работы пользователя осуществляется клиентским приложением и отделяется от серверных процессов. Он накладывает персональные изменения на полученные с сервера данные. Серверные процессы в свою очередь также разделяются на ряд групп:

(1) процессы вычисления разностей контекстов и по ним окрестностей каждого контекста;
(2) процессы кластеризации множества пользователей в окрестностях контекстов, выделения базовой и альтернативных версий

интерфейса в контекстах, и построение стратификации активных в окрестности пользователей;

(3) процессы формирования основы для подсказок;

(4) процессы анализа истории работы пользователей и уточнения их авторитетности;

(5) процесс распределения системных ресурсов между процессами.

Последний процесс регулирует распределение ресурсов между остальными группами процессов, работающими асинхронно и автономно. Процессы не имеют конкуренции по записи — каждый пишет свои выходные данные. Это позволяет вести доработку системы без приостановки сервиса.

Логическая согласованность обеспечивается

- пометкой версии пользовательских изменений единым временем,
- пометками результатов обработки единым для группы процессов временем пользовательских изменений,
- непосредственным доступом к нужным версиям.

Таким образом, решение поставленной задачи предполагает существенно новый подход к организации исполнения. Более подробно этот подход описан в [6].

Список литературы

[1] Huberman B. A., Golder S. A. *Usage patterns of collaborative tagging systems* // Journal of Information Science, 2006. **32**, no. 2, p. 198–208 ↑[]

[2] Xiaohua Hu, Jung-ran Park, Caimei Lu *User tags versus expert-assigned subject terms: A comparison of LibraryThing tags and Library of Congress Subject Headings* // Journal of Information Science, 2010. **36**, no. 6, p. 763–779 ↑

[3] Razikin K., Dion H.Chua, Alton Y. K. Chua, Chei Sian Lee *Social tags for resource discovery: a comparison between machine learning and user-centric approaches* // Journal of Information Science, 2011. **37**, no. 4, p. 391–404 ↑[]

[4] Знаменский С. В. *Ретроспективная основа совместной реорганизации сложных информационных ресурсов* // Электронные библиотеки: перспективные методы и технологии, электронные коллекции // Труды XIII Всероссийской научной конференции RCDL-2011. — Воронеж : Воронежский госуниверситет, 2011, с. 93–101 ↑[]

[5] Демидов А. А. *Проектирование распределённых систем обработки объектных структур данных* // Электронные библиотеки: перспективные методы и технологии, электронные коллекции // Труды XII Всероссийской научной конференции RCDL'2010. — Казань : Казанский университет, 2010, с. 441–447 ↑3

[6] Знаменский С. В. *Процессный подход к эволюционированию информационной системы* // Программные системы: теория и приложения, 2011. **2**, № 4(8), с. 115–125 ↑3

S. V. Znamenskij. *Architecture of collaboratively-changeable hierarchical structure.*

Abstract. A new approach under development to collaborative navigation on a large collection of heterogeneous resources, to be carried out by a large number of users. An important difference from conventional Crowdsourcing is the complete decentralization of management.

The approach is based on the selection and bringing together income of like-minded users to the classification of the collection. Thinking outside the box a user spends adequate efforts on building individual taxonomy, accurate within its immediate interests. The more active user in the interface and the more he has active in same contexts like-minded persons, the more faster and deeply taxonomy will be adjusted for his mind.

The robots that analyze the resources can participate among the users.

The general architecture of the future system and a few algorithms are presented in the paper.

Key Words and Phrases: collaborative software, IS architecture, catalogue of resources, taxonomy, collaborative tagging.

Образец ссылки на статью:

С. В. Знаменский. *Архитектура коллаборативно-изменяемой иерархической структуры* // Программные системы: теория и приложения : электрон. научн. журн. 2011. № 4(8), с. 115–126. URL: http://psta.psiras.ru/read/psta2011_4_115-126.pdf