Metering and Licensing of Resources: Kala's General Purpose Approach

by Sergiu S. Simmel and Ivan Godard

Review by Seung Jun Lee

ABSTRACT

"This paper describes the licensing and metering capabilities of Kala, a persistent data server. Kala offers a suite of low-level primitives for constructing both simple and sophisticated licensing and metering models. Kala allows the licensing and/or metering of access to any software facilities, both data and executable code and associated services."

Outline

- Kala Technical Brief
- Definitions
- Acquiring a Resource –The Basic Algorithm
- The Architecture of main activities of the resource model
- The Definition Activity
- The Cookie Exchange Protocol
- Conclusions
- Questions

Kala Technical Brief

- Persistent Data Server managing distributed, shared, arbitrarily complex and evolving persistent data
- Highly efficient and secure
- Manages the visibility of persistent data elements to its clients,

  Supporting any types of transactions, versions, access control, security, configurations
- Not restrict to any particular model
- Usable as either a link library communicating to a server or as a standalone. Compact and simple
- Used for application such as: kernel of DBMS products, substrate for extended file systems, implementation of language persistence, data manager for groupware applications, text databases, financial distributed transaction systems etc

Definitions

- Metering (“Pay-per-use”)

  An arrangement whereby the consumer pays the producer for as much of the producer’s software as the consumer’s software actually uses

- Licensing (“Pay-per-user”)

  An arrangement whereby the consumer’s software is permitted to use the producer’s software for a period of time in return for a fixed fee, independent of whether or how much the consumer’s software actually uses the producer’s software

- Resource

  An abstraction representing access to a software component

Acquiring a Resource: The Basic Algorithm

- Step1: For each of X's sub-resources Y;
  - 1.a Attempt to acquire Y.
  - 1.b If successful, then go to next sub-resource, if any.
  - 1.c If Y requests a charge, and there is a license from Y to X or any parent of X, then acquire Y with license.
  - 1.d If Y requests a charge, and there is no license for Y, then accumulate charge.
- Step2: Add local added value to a accumulated charge.

Acquiring a Resource: The Basic Algorithm

- Step3: If X's parent has a license for X, then
  - 3.a Accept provisional charges from all X's sub-resources, and remember to pay all charges to X' sub-resources from X's account.
  - 3.b Return success.
- Step4: If X's parent has no license for X, then propose to charge accumulated charge to parent.

Acquiring a Resource: The Basic Algorithm

- Step5: If X's parent accepts provisional charge, then
  - 5.a Accept provisional charges from all X's sub-resources, and remember to pay all charges to X's sub-resources from the actual charge received from X's parent.
  - 5.b Return success.
- Step6: If X's parent refuses the provisional charge, then return failure.

The Architecture

The resource management model involves three distinct activities and related sets of objects:

- The Definition Activity
- The Resource Acquisition and Charging Activity
- The Accounting Activity

The Definition Activity

"Definition" – the activity by which resources are defined, licenses are installed, and meters are filled with meter currency.

Common properties:

- All involve interactions between a customer and vendor.
- All reflect cash transactions between the customer and the vendor.
- All must be totally safe – "**Cookie exchange protocol**"

The Cookie Exchange Protocol

- A mechanism designed to insure the safe and unforgeable definition of resources, installation of licenses and filling of meters at the customer sites
- The mechanism is based on the exchange of "magic numbers" between vendors and customers. –"**Cookies**"

For each definition action, two cookies are involved:

- The Customer Cookie
- The Vendor Cookie

The Cookie Exchange Protocol

- The Customer Cookie
  - generated by the customer on the customer computer and transmitted to the vendor.
  - encodes information that makes it unique, reflecting this particular customer site
  - identifies the customer uniquely throughout the subsequent activity.
- The Vendor Cookie
  - generated by the vendor in response to a customer cookies and to information the vendor has about the associated cash transaction.
  - encode the originating customer cookie, the vendor's identity, and action performed on the basis of customer payment.
  - generated by vendor, using a unique per-vendor copy of the NewCookie program.

The Cookie Exchange Protocol

The cookie exchange protocol is safe against fraud as long as:

- The vendor's NewCookie program is safeguarded by the vendor.
- The customer's site is able to generate customer cookies that uniquely identify the site.
- Valid cookies from NewCookie only work on sites which generated the original customer cookie.
- Invalid cookies don't work at all

Conclusions

For Kala's resource management functionality, Kala meets some requirements. – "**Be Safe**"

- The resource model requires components to be accessible only via their resource objects
- Kala provides a secure storage of data, whereby applications have full control over who can see what and when, via the data visibility primitives
- The dual cookie device permits rights to be communicated between vendors and customers without the need for expensive or cumbersome communication safety provisions.

Question & Answer

- Which is better?
  Pay-per-use ("metering") or Pay-per-user ("licensing")
- Does this license administration mechanism perfectly display efficiently for both customer and vendor?