

Security & Workflow Patent Summary – A Unification of Access Management & Process Control

Business Perspective

A modern business application should consider collaboration as one of its responsibilities. Many business transactions require approval, advice or notices. If the application has information about the people, their relationships and a simple set of rules, it should be able to route transactions based on company policy and ensure that people are only able to see appropriate data. Doing that without intensive security administration has been so difficult that many applications have very limited capabilities.

This patent provides a mechanism that both controls access to data and supports a robust workflow system with very little administrative overhead. Instead of specific assignments of access, the system stores a small set of rules about who should have access to information. These simple rules can address both access and workflow. Because the rules are dynamic, they immediately grant or withhold access as an organization changes.

Background

In SynchSource's Open Framework, a user's access is governed by a set of rules and a dynamic computation of roles based on relationships. Whether or not a user has access to some information on a subject (person, company, pay process) is controlled by the role the user has with that subject. Within a Company, a user might be looking at a subordinate, a peer or their boss. The roles might be Direct Manager, Peer and Subordinate, respectively. In a large system such as SynchSource, which contains over 8,800 information presentations, each with several variations, it is more practical to have rules about access to information rather than access controls to individual pages. Having rules about access to information, allows the system to determine whether a user has access to a page or portion of a page by determining the access to the information on the page.

This approach has several administrative advantages over access control schemes based on page access:

- When several information units are part of a presentation, an independent decision can be made as to whether or not each part should be presented.
- When an information unit is constructed or changed, the policy about that information can be explicitly stated and enforced, removing the possibility that inadvertent programming changes will violate security policies.
- When an information unit is constructed or changed, the mechanism guarantees that the same access rules will be applied wherever the information unit appears.

The rules:

An access rule is projected as a simple triplet:

User A has Role B for Subject C.

Information units have role associations:

Information Unit AXXQQYY can be accessed by Roles A, B, and C.

If all Information Units are associated with their allowed access roles, it is easy to determine whether or not a User has access to some Information Unit.

When such triplets can be manipulated by relational logic, we can take the above assertion, and ask, Who has Role B for Subject C? And, Who has access to a particular Information Unit? If we have the above assertions, then when asking the question, Who has access to Information Unit AXXQQYY? We can answer that: User A has access to Information Unit AXXQQYY.

The SynchSource Process Management Facility can always inspect the next steps in a Process to determine what Information Units are required and consequently whether the current User has access to those resources. This is done by the operation:

- Does Current User have an access role for the subject that provides access to the required Information Unit?
- If the answer is no, a further operation from the same triplet can determine:
What User has access to the required Information Unit?

The simple triplet allows the Process Manager to transform a simple stepwise process into a workflow between multiple users who have the required access.

Example of the mechanism:

Rule Self:

A User should have the role Self:

if they are accessing their own records
and they are currently employed by a company
and have an employment status of A-active, L-leave without pay, or P- leave with pay

Rule Mgr:

A User should have the role DMgr (Direct Manager) for a Person:

if the User currently occupies a position
and that position currently manages another position
and the Person currently occupies the second position

Using the two rules above and a Process that has four steps with associated Information Units:

Leave Request Form – Information Unit = Leave Request
Store Proposed Form – Information Unit = Proposed Leave Request
Approve Leave Request – Information Unit = Approving Leave Request
Store Approved Form – Information Unit = Approving Leave Request

and Access rules:

Leave Request – Direct Manager, Self
Proposed Leave Request – Direct Manager, Self
Approving Leave Request – Direct Manager

Then if a person creates a Leave Request for themselves, the process flow might look like:

- Self – Leave Request Form
- Self – Store Proposed Leave Request (Routed to Direct Manager)
- Direct Manager – Approve Leave Request
- Direct Manager – Store Approving Leave

On the other hand, if the Direct Manager were to create the Leave, then the process would look like:

- Direct Manager – Leave Request Form
- Direct Manager – Store Approving Leave

In this example, the workflow is dynamically determined from the access rules because the rules can be inverted – User has Access and Who has Access.

The essence of the “Access Control and Workflow Routing by Real-time Relationships – a structure and process for access control and workflow routing in online applications” patent is to construct Access Rules in the form of the above triplets and thereby control access. The same triplets are used to construct a routing for a workflow.

Structured SQL

A further provision of the patent is that the triplets be constructed as structured SQL statements. This provision means a relational database system can be used to efficiently compute both access patterns and routings.

The structure requires that these SQL statements produce relations with at least three specific columns: UserId, SubjectId, and RoleName. The UserId is the key of a record that defines the process’ current User. The SubjectId is the key of the Subject of the Process (Person, Company, etc.). The RoleName is the role that the User has for the Subject.

Examples:

Rule Self:

- A User should have the role Self:
 - if they are accessing their own records
 - and they are currently employed by a company
 - and have an employment status of A-active, L-leave without pay, or P-leave with pay

SQL:

```
Select P.PersonId as UserId, P.PersonId as SubjectId, 'Self' as RoleName
from person p
where exists
(select personid from Person_employment pe
where p.personid = pe.personid
and current_date <= pe.enddate and pe.effectivedate <= pe.enddate
and (pe.emplstatus = 'A' or pe.emplstatus = 'L' or pe.emplstatus = 'P'))
```

Rule Mgr:

- A User should have the role DMgr (Direct Manager) for a Person:
 - if the User currently occupies a position
 - and that position currently manages another position
 - and the Person currently occupies the second position

SQL:

```

SELECT p.personid AS userid, pp.personid AS subjectid, 'DMgr' AS rolename
FROM pers_pos p
JOIN pos_pos p_p
ON p.positionid = p_p.positionid
AND current_date <= p_p.enddate
AND current_timestamp >= p_p.createts AND current_timestamp <= p_p.endts
JOIN pers_pos pp
ON p_p.topositionid = pp.positionid
AND pp.persposrel = 'Occupies'
AND current_date <= pp.enddate
AND current_timestamp >= pp.createts AND current_timestamp <= pp.endts
WHERE p.persposrel = 'Occupies'
AND current_date <= p.enddate
AND current_timestamp >= p.createts AND current_timestamp <= p.endts

```

These SQL statements can be provided directly to a relational database system or can be stored as compiled views.

It is a property of relational systems that the result of any SQL statement is a relation and that such relations can be further operated upon by SQL to produce another relation.

If we take the Self SQL and submit it with UserId = 1 and SubjectId = 1, it will create a relation of the form:

UserId	SubjectId	RoleName
1	1	Self

If we submit the Self SQL with UserId = 1 and SubjectId = 2 if it returns no rows.

If the Direct Manager for User 1 is User 2 and we submit SQL with UserId = 2 and SubjectId = 1:

UserId	SubjectId	RoleName
2	1	DMgr

Thus we know that UserId 2 has the RoleName DMgr when accessing information for SubjectId 1.

In the above process flow, to determine which user can perform the Approve Leave Request process step, we can submit a query with SubjectId = 1 and RoleName = DMgr and will receive the result:

UserId	SubjectId	RoleName
2	1	DMgr

Thus we know that UserId 2 can access the Approve Leave Request since that User has RoleName DMgr when accessing SubjectId 1.

In this example, the same rules that determine access also determine process routing. This is possible because the rules are expressed as SQL and will provide RoleName when queried with UserId and SubjectId or will provide UserId when queried with SubjectId and RoleName.

Summary

Since consistency between access control and process management is essential, having a single mechanism that consistently drives both functions eliminates inconsistency.

Storing role definition rules as SQL views allows roles to be dynamically computed for each process step. Since the data in a transaction system is constantly changing it requires dynamic role determination:

- A manager should have access to his people's information as soon as his promotion is approved and access to a person's information as soon as a transfer to them is complete.
- Likewise a manager should not have access to information about their prior subordinates when their transfer to another department is complete.

These requirements can be met with this mechanism with very little role administration. If access rights and roles had to be entered into a transaction system, the rights and roles would be continuously incorrect.

The sheer volume of stored access control rights prohibits their use for anything but the simplest systems. Imagine a system containing 100,000 people. Each person might have access to 100 different pages of information. This structure leads to 10,000,000 access controls. Such volume guarantees errors.

A dynamic system contains no access control rights, but rather a small number of rules that can control both access and process flow.

About SynchSource

SynchSource Inc. has developed a suite of open source HR, workforce management, benefits, and payroll solutions that are rapidly establishing the company as a leader in the service bureau software marketplace. SynchSource's fully integrated platform provides Human Resource Outsourcers (HROs), Professional Employment Organizations (PEOs), Payroll Outsourcers, Benefits Administration Outsourcers, and other service providers with a SaaS delivery model designed to empower their clients, accommodate unique requirements without customizations, reduce overall administrative support costs, and provide more value-added services at improved margins. The patented time-relational data model and the patented workflow and security engine are at the core of the SynchSource platform.