Multi-tenancy in Cloud-based Collaboration Services

David Banks, John Erickson, Michael Rhodes

HP Laboratories
HPL-2009-17

Keyword(s):
FRACTAL, Multi-Tenanted, Architecture, Cloud, SaaS, HP

Abstract:
In this paper, we argue that increased outsourcing of non-core competencies will drive the demand for a new generation of multi-tenanted cloud-based platforms that address the needs of content-centered collaboration between organizations. We introduce the FRACTAL conceptual prototype which has allowed us to evaluate the suitability of current enterprise content management (ECM) technologies for this type of platform. Our early results highlight several areas, particularly around multi-tenancy, where we feel current platforms are inadequate and fundamentally new approaches are required.

External Posting Date: February 21, 2009 [Fulltext] Approved for External Publication
Internal Posting Date: February 21, 2009 [Fulltext]

Multi-tenancy in Cloud-based Collaboration Services

David Banks
Hewlett Packard Labs
Longdown Avenue, Stoke Gifford
Bristol, BS34 8QZ, UK
+44 117 3128244
dbanks@hp.com

John S. Erickson
Hewlett Packard Labs
PO Box 1158
Norwich, VT 05055 USA
+1 802 649 1683
john.erickson@hp.com

Michael Rhodes
Hewlett Packard Labs
Longdown Avenue, Stoke Gifford
Bristol, BS34 8QZ, UK
+44 117 3128173
michael.rhodes@hp.com

ABSTRACT

In this paper, we argue that increased outsourcing of non-core competencies will drive the demand for a new generation of multi-tenanted cloud-based platforms that address the needs of content-centered collaboration between organizations. We introduce the FRACTAL conceptual prototype which has allowed us to evaluate the suitability of current enterprise content management (ECM) technologies for this type of platform. Our early results highlight several areas, particularly around multi-tenancy, where we feel current platforms are inadequate and fundamentally new approaches are required.

Categories and Subject Descriptors

H.3.5 [**Information Storage and Retrieval**]: Online Information Systems – *Commercial services, Web-based services*

General Terms

Management, Performance, Design, Reliability, Security

Keywords

FRACTAL, Multi-Tenanted, Architecture, Cloud, SaaS, HP

1. INTRODUCTION

As we approach the end of the first decade of the 21st century, we are witnessing a disruptive change in the provisioning of information technology: the advent of the era of cloud computing. In an increasingly global marketplace, businesses are seeking to operate more efficiently by outsourcing non-core competencies. There are two sides to this trend. First, for most organizations information technology is not a core competence and is something they would prefer to leave to specialists. Until recently, the only option for these companies was to have those specialists on-premises, but offerings from the likes of Google, Amazon and Salesforce.com are becoming increasingly viable alternatives. Second, businesses often choose to excel in a single area and partner for the rest. Collaboration across organizational boundaries is now a core part of the product development process, yet traditional Enterprise Content Management (ECM) software has not kept up, leaving users in collaborating organizations to use email as their lowest common denominator. Over the next few years, as the millennial generation starts to enter the workforce, we expect them to act as catalysts of change, increasing the pace of adoption of new tools.

In response to these trends, we envision a new generation of cloud-based collaboration platforms emerging to address the needs of content-centered collaboration between businesses. Although superficially similar to the best of today's ECM systems, these platforms will operate on a massive scale, simultaneously supporting thousands of organizations and millions of users. The FRACTAL project [1] in HP Labs has the long term goal to design and deploy such a platform.

In this short paper we first introduce the FRACTAL Conceptual Prototype, built using Alfresco™, a leading open source ECM system. We then present our rationale for why neither Alfresco nor any other leading ECM system is a suitable base for such a cloud platform.

2. FRACTAL CONCEPTUAL PROTOTYPE

2.1 Goals of the Prototype

We had three distinct goals for the prototype: first, we wanted a functioning system that would help us to better envision FRACTAL from an end user perspective; second, we wanted to clarify requirements for the underlying platform; and third, we wanted to understand limitations of current ECM technologies for realizing multi-tenanted cloud-based applications.

2.2 Key Features

The key features we wished to demonstrate were:

Content Spaces: hosted spaces that bring together people, content, collaborative tools, and customizable active behaviors.

Active behaviors: a way for end users to define functional extensions operating within the context of a content space involving content, metadata, automated processing services and tasks carried out by other users. An active behavior may be manually invoked as needed, or it may be automatically triggered by a change to a content space or the passing of time. An invocation may involve a single content object or many objects in parallel. Their complexity ranges from automatically creating up-to-date PDF versions of documents as they are modified, to running workflows to automatically collate information from several collaborating organizations into a single document.

Agile configuration: must be light-weight, low-touch and customizable by end users without IT involvement.

Open and extensible by third parties: an Internet platform with open APIs, where third parties are motivated to develop customizations/extensions that can then be published through a marketplace and easily discovered by end users.

2.3 Technical Approach

We evaluated several technologies as a starting point for the prototype, including Joomla, Drupal, Alfresco, Liferay, TikiWiki and SharePoint. Our selection criteria included: strong document management features; embedded workflow; social capabilities (blogs, wikis, tagging); and user interface qualities similar to those we envisioned for FRACTAL. We selected Alfresco's new Share technology [2] because it excelled across all these factors.

2.4 Conceptual Prototype Overview

In this section we give a brief tour of the prototype. For further details, we refer the reader to a series of short demo videos we have produced based on the prototype [3].

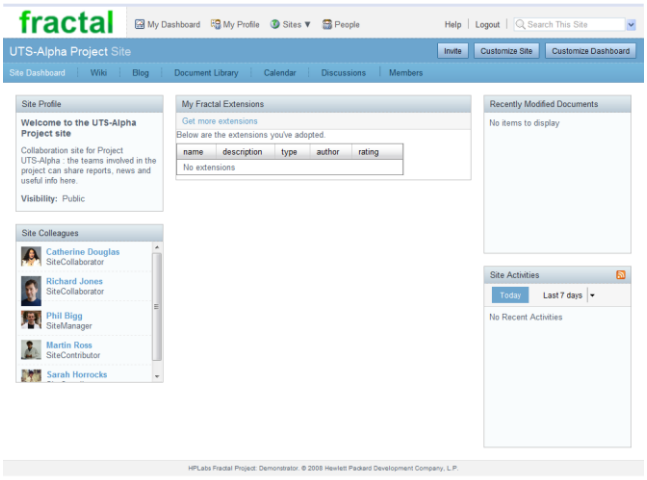

Figure 1. A FRACTAL Content Space.

In Figure 1 we show an example content space to support a collaborative pharmaceutical research project called *UTS-Alpha*. The content space has a customizable set of collaboration tools (wikis, blogs, etc) and a configurable default view, the dashboard.

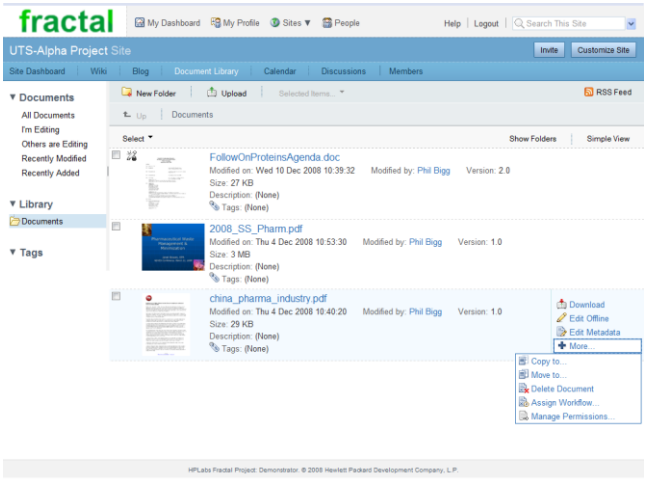

Figure 2. The Document Library Application.

In Figure 2 we show one of the collaborative applications included in the content space: a document library providing versioning and concurrent editing capabilities. Thumbnails and previews are generated automatically, using content transformation services available within the platform.

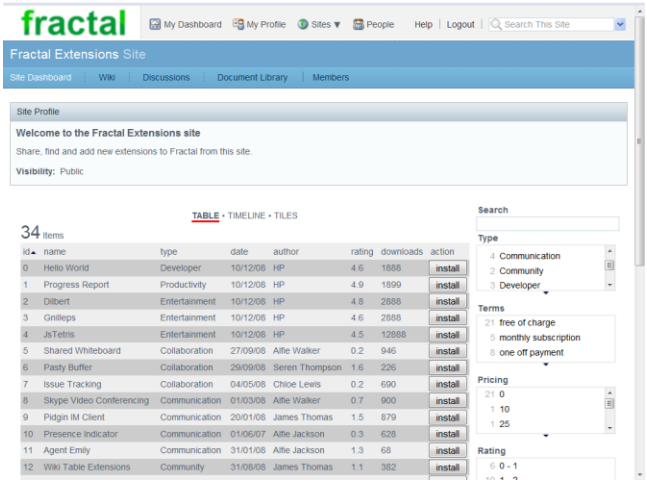

Figure 3. The FRACTAL Extensions Marketplace.

In Figure 3 we show the FRACTAL Extensions Marketplace, a special-purpose content space that provides a place for developers to publish extensions and a rich set of search and browse capabilities to enable users to discover them. In the prototype we used the Simile Exhibit faceted browser [4], which gave us several views (tabular, timeline, etc). In the figure, different selection facets are visible, derived from descriptive, commercial and social metadata bound to the listed extensions. Adopting an extension into a content space simply requires clicking the install button, not unlike adopting Gadgets for the iGoogle interface.

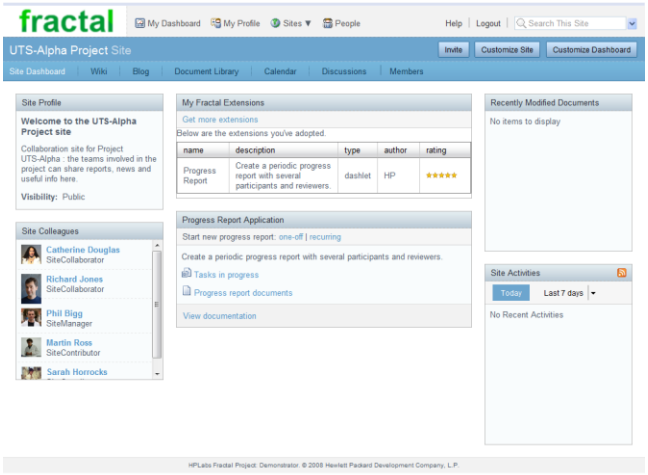

Figure 4. The new Progress Report Application.

In Figure 4 we show the effect of adopting the *Progress Report* application into the UTS-Alpha content space; a new dashboard component (dashlet) has been added and is ready to use. The *Progress Report* application is an example active behavior we developed, implemented mostly as a jBPM workflow, that facilitates periodic authoring of a report by members of the content space, with the completed report being automatically archived in the document library.

3. TECHNICAL CHALLENGES

We successfully used Alfresco Share to rapidly prototype a functioning system with many of the end-user characteristics we envision for FRACTAL. Our success notwithstanding, we believe neither Alfresco nor any of the other leading ECM platforms

provide a suitable base for a multi-tenanted, cloud-scale collaboration platform. Many of our concerns pertain to multi-tenancy, thus we start with the definition of multi-tenancy put forward by Bob Warfield [5]:

Multi-tenancy refers to the ability to run multiple customers on a single software instance installed on multiple servers. This is done to increase resource utilization by allowing load balancing among tenants, and to reduce operational complexity and cost in managing the software to deliver the service. From a customer's perspective, multi-tenancy is transparent. The customer seems to have an instance of the software entirely to themselves. Most importantly, the customer's data is secure relative to other customer's data and customization can be employed to the degree the application supports it without regard to what other tenants are doing.

Problems with this definition arise when trying to apply it to a service that facilitates collaboration between organizations. The hard segmentation called for between customers' data actually makes it impossible for different customers to share data when they need to collaborate. To overcome this, we argue "tenant" and "customer" should not be thought of as synonymous. In a multi-tenanted collaboration service, a tenant should instead be thought of as a collection of distinct, collaborative activities and related content – a *content space* in FRACTAL's terminology. A content space may have a single user and thus be a private space. It might instead have multiple users from the same company, and thus be a collaborative company space. Finally, it may have users from different organizations, and be a collaborative space for inter-organizational activities.

3.1 Data Isolation between Tenants

The first type of isolation we consider is data isolation: ensuring that tenants' data is kept adequately segregated, and that tenants are not able to gain unauthorized access to each other's data.

In Alfresco Share, all content spaces (*sites* in Alfresco) are persisted to a single shared store.[1] This store is backed by a relational database, together with a file system that contains content and a Lucene index. The use of a single shared store is unsatisfactory for a couple of reasons.

First, because there is a single index spanning all content spaces, query times increase with the number of content spaces, as well as with the size of the content spaces.

Second, as files (representing content) from different content spaces share the same file system directories, it becomes impossible to perform file system-level backups that segregate tenants' data onto different sets of media. Not only are there legal implications to this, but it also prevents the service provider offering tenants copies of their own backup media.

3.2 Application Isolation between Tenants

In addition to data isolation, it is important that customizations or functional extensions to a content space are not visible by default to other content spaces. For example, if an application has been adopted into one content space, parts of that application (e.g. dashlets) should not be visible to users of other content spaces.

In Alfresco Share, applications are implemented as web scripts written in a combination of server-side Javascript and Freemarker templates. However, all content spaces (*sites* in Alfresco) share a common search path for web scripts, so when a new application is introduced into one content space it is actually available to all content spaces. Another way the functionality of Alfresco can be extended is with custom jBPM workflows, but these are also deployed globally and suffer from the same lack of isolation.

Application isolation requires functional extensions to be managed more like data objects within content spaces, rather than in a separate global space, as is currently done within Alfresco.

3.3 Performance Isolation between Tenants

A third type of isolation ensures resource-intensive activity in one content space does not impact the use of other content spaces. This is one of the hardest architectural challenges when designing a multi-tenanted service because it directly conflicts with the goal of reducing service costs by sharing resources between tenants. In general, a multi-tenanted service should adopt several approaches to minimizing the impact of tenants on each other.

First, the service should track resource usage on a per-tenant basis. Resource usage typically includes storage, I/O bandwidth, CPU usage, and possibly memory usage. Such tracking enables resource intensive tenants to be identified. It is also worth tracking resource usage against other dimensions, such as per user, per organization, and per application. The latter allows poorly written applications to be identified and possibly blocked or throttled until improvements are made.

Second, tenants should be charged based on resources consumed. This form of pricing (as opposed to a flat rate or fixed subscription) serves as a form of feedback to make users more sensitive to what they are doing. All of the large-scale cloud platforms (Amazon Web Services, Google App Engine, etc.) utilize some form of resource-based pricing.

Third, the service should dynamically load balance tenants across hardware resources. Usage patterns are likely to be bursty and there will be times when resources are over-allocated causing hot spots to develop. The impact of such hot spots can be minimized by dynamically altering resources assigned to tenants.

Fourth, a tenant's activities can be throttled. This is a last resort because repeatedly throttling a tenant will likely discourage future use of the service.

The type of fine-grained monitoring, management and billing infrastructure necessary to support these approaches is absent from current ECM platforms.

3.4 Tolerant of Hardware Failures

In a cloud-scale service provisioned across thousands of servers, disk and server failures will occur routinely and must not result in loss of service. In addition, continuous hardware upgrades must be possible without interrupting the service to any tenants.

In general, ECM platforms use a variety of techniques to support high-availability deployments. In Alfresco, servers can be clustered and share state using a transactional object cache. A single database is shared between servers, which can itself be clustered. Indexes are maintained locally, loosely synchronized to the object cache. Finally, content is stored either on a single shared file system or on local file systems that are replicated [8].

This approach to achieving high availability is expensive in terms of hardware, software licenses and operational costs. It also does not scale to a very large number of nodes. A cloud-scale service supporting thousands of tenants would require many independent

[1] We note Alfresco do implement a form of multi-tenancy in some of their products [6], but not currently in Alfresco Share [7].

clusters (pods), simply shifting the problem of load balancing to a different level.

3.5 Per-Tenant Levels of Service

In a multi-tenanted service, different tenants may require—and be willing to pay for—different levels of service. For example, one tenant might place a very low value on their data since they maintain their own backup and would prefer the cheapest service possible. Another tenant may want their data to be replicated multiple times, across data centers in multiple continents. With ECM platforms, decisions about degrees of redundancy and replication are deployment decisions that apply to the service as a whole, and cannot easily be varied per-tenant.

3.6 Ease of Extension by Developers

In FRACTAL, we seek to create an open platform that can be readily extended by third party developers. There are a number of factors that make a platform attractive to developers. These include: being able to code in a familiar programming language; having well designed, stable APIs; providing accurate documentation; having effective frameworks for testing and logging; and an integrated development environment.

Our experience is that it is not easy to develop new extensions for commercial ECM platforms. Some ECM companies have not set out to create such open platforms in the first place. In addition, few commercial ECM platforms are sufficiently widely deployed to have attracted a large following of third party developers.

3.7 Ease of Customization by End Users

With FRACTAL we want to empower ordinary users to tailor content spaces to their needs. We want their customizations to extend beyond simply adopting applications written by professional developers; rather, we want to create an environment where end users are able to author their own extensions that precisely meet their needs and, if appropriate, share these with the broader community.

Many ECM systems embed simple scripting and workflow capabilities that, in theory, provide an easy route to authoring simple extensions. Alfresco, like many other ECM systems, embeds the JBoss jBPM workflow engine to allow custom workflows to be developed. In our prototype we evaluated the suitability of this environment for non-technical end users. Unfortunately, our results were not positive. Even for experienced software developers, implementing the jBPM workflow for the *Progress Report* application described in Section 2.4 was very time consuming. We found several impediments: the workflow design environment was Eclipse-based rather than an integral part of the ECM platform; the graphical editor gave only a partial view of the workflow, requiring actions to be authored in code; users needed a good understanding of concurrent programming concepts, such as forking and joining; XML configuration changes were needed elsewhere in the platform to support the workflow, such as defining custom content models and changes to the Spring startup configuration. This complexity needs to be eliminated if non-technical users are to have a chance at authoring their own custom active behaviors. The environment needs to be as simple to use as an Excel spreadsheet.

4. RELATED WORK

In the scientific domain, the myExperiment work [9] provides an excellent proof point that users can successfully author and share complex workflows, given the right tools.

The Ning social network platform [10] demonstrates how easy it should be for users to create their own customized spaces. The key difference between Ning and FRACTAL is that Ning is consumer focused and lacks document management and workflow capabilities.

Several cloud application platforms have recently emerged that free developers from worrying about scaling the infrastructure supporting their application if it is successful: Microsoft Azure [11], Google App Engine [12] and Salesforce Force.com [13].

5. CONCLUSIONS AND FUTURE WORK

In this paper, we overviewed the FRACTAL project at HP Labs, and described the FRACTAL Conceptual Prototype. We explored what we see as the key requirements for a multi-tenanted cloud-scale platform focused on content-centric collaboration. We argued the current generation of ECM technologies is not a good match, and highlighted some of the improvements required. Over the next 12 months, our research will focus on alternative implementation patterns to satisfy these requirements.

6. REFERENCES

[1] Erickson J et al 2009. Content-Centered Collaboration Spaces in the Cloud. HPL Tech Report HPL-2009-11. Submitted Jan 2009 to *IEEE Internet Computing* special issue on Cloud Computing.

[2] Alfresco Share: http://wiki.alfresco.com/wiki/Alfresco_Labs_3_Share_Feature_List

[3] FRACTAL Prototype Videos http://purl.oclc.org/NET/hp-fractal-prototype-videos

[4] Simile Exhibit http://code.google.com/p/simile-widgets

[5] Bob Warfield, SmoothSpan Blog 27th October 2007 http://tinyurl.com/8wj78u

[6] Alfresco Multi Tenancy http://wiki.alfresco.com/wiki/MT

[7] Alfresco Repository Roadmap http://wiki.alfresco.com/wiki/Repository_Roadmap

[8] Alfresco Cluster Configuration http://wiki.alfresco.com/wiki/Cluster_Configuration_V2.1.3_and_Later

[9] De Roure, D., Goble, C. and Stevens, R. (2008) The Design and Realisation of the myExperiment Virtual Research Environment for Social Sharing of Workflows. Future Generation Computer Systems. DOI http://dx.doi.org/10.1016/j.future.2008.06.010

[10] Ning Social Network Platform http://ning.com

[11] Microsoft Azure http://www.microsoft.com/azure

[12] Google App Engine http://code.google.com/appengine

[13] The Force.com Multitenant Architecture: Understanding the Design of Salesforce.com's Internet Application Development Platform. http://tinyurl.com/8uxxb7