Globally Distributed BookPrep

Open Cirrus-Hosted Service for Book Preparation

Prakash Reddy,[1] Shariff Dudekula,[2] Susanth Puthanveedu,[2] and Dejan Milojicic[3]
[1]HP Imaging and Printing, [2]HP Enterprise Services, and [3]HP Labs
[1,3]Palo Alto, CA, USA, [2]Bangalore, India
[first.lastname@hp.com]

***Abstract*— BookPrep is a Print-On-Demand service that takes raw scans and converts them to print-ready files. It requires large amount of storage and takes an average of 5 hours of CPU time to process a single book with about 300 pages. The experiment we conducted involved moving the processing of books on Open Cirrus closer to the location of the data. At three Open Cirrus sites we installed BookPrep service and we pre-populated each site with region-specific scanned books. When requests come in to process a book, each request is routed to the compute node closest to the source data. The compute node is then expected to store the processed data on the same network. The compute nodes are allocated and de-allocated based on demand. There is a cloud based metadata repository that is used to update the metadata associated with each book regardless of the location of the source and derived data. The goal of this experiment is to determine if performance can be improved by moving book processing close closer to source data location. The fundamental reason behind the success of MapReduce is the notion of moving compute close to data and we would like to see if that same principal can be applied to a pull based scheduling model.**

Keywords: distribution, Clouds, Web services, and Imaging and printing.

I. Introduction

Cloud computing is ideal for computational tasks with unpredictable demand, where sudden surges of demand cannot be met by local resources. One such example of computational task is processing and printing of books on demand (POD). This service entails taking scanned images as input, applying image processing algorithms to remove artifacts, converting them into a printable version, printing them, and then shipping to the customer as and when a book is purchased. In this paper we will describe BookPrep [1][2], one such online service.

The print-on-demand book service has a web portal which accepts print on demand book orders from users, and scanned book uploads from authors. A typical scanned book with about 300 pages needs 1GB of storage and it takes approximately 5 hours of computation time to process a book on a single node and generates 2GB of processed data. The processed data includes print ready files as well as data used to support online viewing of the processed books. Since the demand for compute resources is unpredictable a dynamic resource allocation mechanism is needed to build a cost effective solution.

The BookPrep service was implemented and has been production at HP for over two years. Initially, it was developed on Open Cirrus site in Palo Alto as a Web service. However, original implementation was working off of one site. Subsequently, the original architecture was modified to support multiple sites the storage and the fulfillment of orders were still handled at a single location. This helped with scaling the compute resources but did little to improve the overall throughput.

In this paper, we describe a truly distributed Bookprep, where all of components including computation, storage, and order fulfillment are distributed across multiple sites. The goal of this effort is to further increase scalability and also to reduce costs (bandwidth, shipping), by enabling regional access to BookPrep processed content. For this purpose we have leveraged three Open Cirrus sites: Palo Alto (original site), GaTech, and Karlsruhe Institute of Technology. We are also in the process of installing on CMU, MIMOS, and ETRI sites, but we did not manage to complete installation to report in this paper.

Figure 1 demonstrates the concept of the globally distributed BookPrep. Users submit requests from anywhere in the world to a single portal. The portal redirects requests by selecting optimal location where the book will be processed and then shipped to the customer. In this process the most communication takes place between the stored book and computational unit. For that reason the books are cached on the regional boundaries. The next costly component (both in terms of time and money) is shipping of processed copy of physical book. In this paper, we used collected data from three sites to estimate scalability, performance and costs of three different architectures.

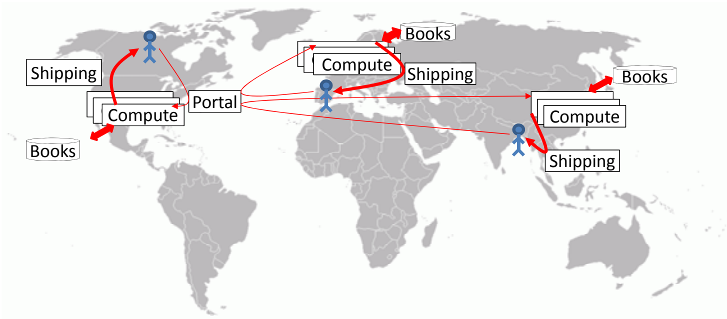

Figure 1. Concept of Distributed BookPrep. Small amount of communication is handled across geographies and as much as possible locally.

The rest of the paper is organized in the following manner. Section II describes BookPrep original service and Section III the first version of distributed BookPrep implementation. In Section IV, we present the globally Distributed BookPrep architecture. In Section V we discuss evaluation of our work. Lessons learned are presented in Section VI and future work in Section VII. In Section VIII, we compare our approach to related work and in Section IX we provide a summary.

II. BookPrep

BookPrep is a tool for processing scanned versions of books into generating Print on Demand (POD) ready files and online viewable files (http://www.bookprep.com). It was originally developed in 2007 and it went into production in 2009. Over its lifetime it went through a few architectures, three of which we discuss and compare in this paper.

Automatic Book Preparation

A vast majority of the out of print and out of copyright books are in libraries and primarily collecting dust. Fortunately, efforts are underway by the Open Content Alliance, Microsoft, Google and others to digitally scan these books, bringing them online. These books are scanned using high resolution photography, to avoid destroying the originals. However, this photographic scanning produces page images that are not directly useful for print because of lighting, alignment, scan artifacts, as well as aging and wear and tear from use.

Preparing books for POD from scanned images involves solving and automating a variety of technical steps, such as detecting page boundaries, adjusting for lightning, removing background noice, deskewing, text sharpening and formatting into a form that is readily acceptable to a print service provider (PSP).

One of the key problems in preparing scanned images for printing is detecting page boundaries on the scanned images and finding a crop region that can be applied to all the pages in a book. In addition we detect the content boundary for each page which helps in eliminating lot of the artifacts introduced during the scanning. The Figure 2. below shows images before and after page margin detection.

Figure 2. Page margin detection

Once the pages are cropped, we run the images through a set of enhancement algorithms, where each image is deskewed if necessary, text is sharpened, and images are adjusted. Color conversion also happens at this stage. Figure 3 shows examples of original images and processed images after color/illumination correction and text sharpening.

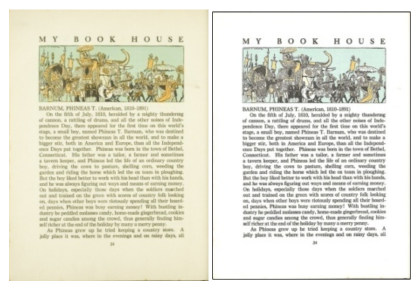

Figure 3. Color/illumination correction and text sharpening

Once the pages are cleaned they are packaged into formats that can be printed by any of the Print Service providers in the network. The Print Service Providers (PSPs) are designed to take print ready files print one copy at a time and also handle the order fulfillment. One of the gaols of this work is to leverage the PSP network to minimize the cost of shipping by directing a sale/fulfillment to a PSP that is geographically closest to the customer ordering the book.

BookPrep is one of the embarrassingly parallelized applications with clear delineation of computation, storage and networking, almost ideal for Cloud computing. In its lifetime, it has processed close to a million of book copies, consuming close to 5 million CPU hours (single core), and approximately 450TB of storage. It is in a daily production use, 24x7.

It is implemented in Java and C++, in approximately 30,000 lines of code. Originally it started on open Cirrus partition in Palo Alto, but it has since expanded on a dedicated cluster, as well as onto Amazon Web Services [3] and other Cloud providers.

The original BookPrep implementation (see Figure 4) consists of several components each having different bandwidth, storage and compute requirements. The components include a) Process Queue (PQ) b) Content Acquirer (CA) c) Process Pipeline (PP) d) Formatters (for printing and online viewing) and e) Content Delivery (CD) for Print Service Providers (PSPs) and online viewing.

A bank of in-house compute nodes acquires, processes, formats and transfers print ready books to PSPs. The number of books that can be processed at any given time is a function of the number of CPU cores available. The queue service and support for online viewing is handled by a Web Service (WS).

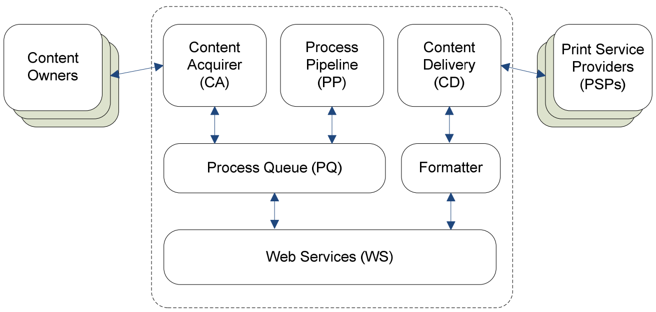

Figure 4. Original BookPrep Implementation.

III. DISTRIBUTED BOOKPREP ARCHITECTURE

This architecture does not support dynamic allocation of compute resources needed to meet peak demand. Figure shows the components of the distributed BookPrep implementation. The interaction between Compute nodes and the Process Queue is through REST APIs. This allows us to support multiple different cloud infrastructures. The content owners and the raw scan data could be hosted on different clouds. The interaction and the mechanisms used to acquire content are specific to each content owner and are encapsulated in the Content Acquirer.

The Queue Interface (QI) on each processing node is responsible for talking to the queue and acting on the responses. The processing pipeline is the largest consumer of CPU that takes the raw scans and prepares them for POD. Once the processing is complete, the processed file is transferred to the resident storage infrastructure. The processed data is packaged and uploaded to the appropriate PSP when orders are processed. Depending on the PSP the data could be transferred once per book or each time a book is ordered.

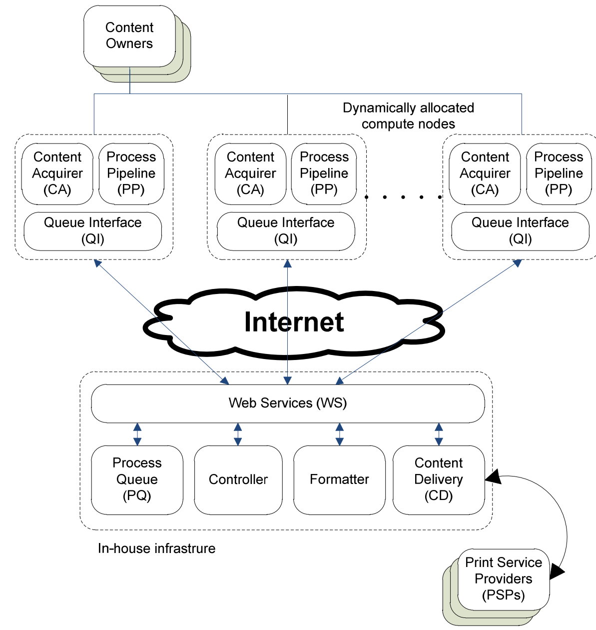

Figure 5. Distributed BookPrep Architecture

IV. GLOBALLY DISTRIBUTED BOOKPREP IMPLEMENTATION

While this architecture does support dynamic scaling of compute resources needed to meet peak demand the storage is still centralized and the bandwidth requirement are quite high. Figure describes a globally distributed BookPrep which entails replicating most BookPrep components, as opposed to simply replicating the image processing portion of the pipeline. In this version, we distribute BookPrep by replicating the initial architecture in global locations and making the portal aware of these locations.

The central web portal hosts the algorithm that determines preferred location and schedules the processing and or fulfillment. The raw scanned images can also be distributed to optimal locations for example the all Korean language books could be hosted on an infrastructure that is physically located in Korea. In the current implementation, for simplicity reasons, authors interact with regional BookPrep instances.

In this new architecture books are processed in the region that contains the source and the processed data is also stored in the same region. The assumption is that the demand for these books is highest in that region and all book orders are fulfilled by the nearest PSP. This methodology allows us to reduce the bandwidth costs since there is no need to transfer the raw data across regions the compute performance is improved since the data is close to the compute farm and finally the costs of shipping are reduced because the book is printed and shipped locally. This is important because often times the cost of shipping far exceeds the price of the book if all books are fulfilled from a central location or by a single PSP.

In the next version, we shall also distribute the process queue and replace it with the distributed algorithm for request fulfillment, with optimized caching and replication.

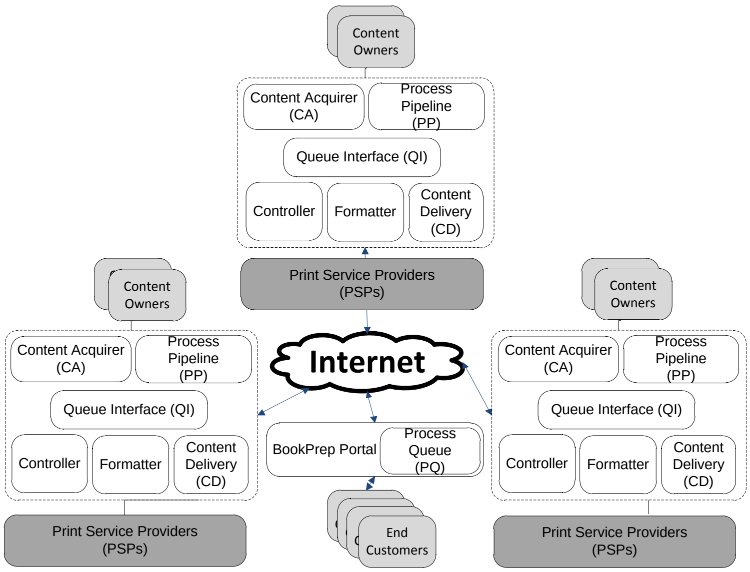

Figure 6. Globally Distributed BookPrep

V. Evaluation

The configuration of each BookPrep site is provided in Table 1 below.

Table 1. Configurations of Open Cirrus BookPrep Instances

Open Cirrus Site	#nodes	Cores	Storage	Networking
Palo Alto	33	448	240T (Ibrix)	1GB/s
KIT	1	16	503T (NFS)	TBD
GaTech	5	12	TBD	TBD

Size of the raw scans for books range from 20 MB to 1GB on an average 400 MB of data needs to be uploaded into repository. Processed content (print ready file) size is 60-70% of the size of the raw scan. A 400MB raw scan book would translate to a 200-250MB processed content data.

On the external machines on an average the bandwidth speed is 1.5 MB/second. Table 2 below gives samples for speed for 3 book sizes.

Table 2. Download Time as a Function of Different Book Sizes

Size (MB)	Download Time (sec)	Speed (MB/sec)
381.50	232.8	1.64
941.29	525.6	1.79
418.19	289.2	1.45

Breakdown for various components of BookPrep in terms of how long individual component takes time to execute is provided in Table 3 below.

Table 3. Breakdown of BookPrep Functions.

Page #	Download time (min)	Process Time (min)			Total time (min)	Avg page process time (min)
		Convert	MC	AEB		
72	1.43	16	23	6	51	1.41
120	2.85	27	43	12	92	1.30

In the table above the following components are described:

- *Convert* is a normalization process that converts a variety of input formats (jp2, jpg, tif, png) and scales the images so the resolution of the input images are consistent.
- *MC* determines the page and the content boundaries and does de-skew operation to clean up the images.
- *AEB* removes the artifacts in images that are introduced during the scanning process or those that occur due to age. This also sharpens the text so the printed text is clear.

There are multiple benefits of deploying and using BookPrep in a globally distributed configuration.

- **Shipping** costs will be reduced because placement algorithm at portal directs requests for fulfillment by a print service provider that is geographically closest to the consumer.
- **Bandwidth**. The bandwidth requirements can be reduced considerably if the books are processed to where the data is stored. However there are times when processed data is transferred to a PSP that is closer to the user which increases the cost of data transfer the hope is that these costs are offset by lower cost of shipping.
- **Computation.** Obvious benefit is in enlarging the number of cores, at a scale that cannot be accomplished at a single site. Second benefit is in leveraging fluctuation of the cost of processing, so that processing can always be done at the cheapest site (assuming that book is also available at that site and transfer/storage costs would not outweigh benefits).
- **Storage.** Replication of storage at different sites obviously increases scalability but it also increases cost. At the large scale the scalability is more important than cost reduction.

Table 4 below compares scalability of three different implementations. Intuitively, the benefits of globally distributed BookPrep consist in scaling and shipping.

Table 4. Scalability Comparison of BookPrep Implementations

Configuration	Compute	Communicate	Ship
Single Site BookPrep	1 site cores	1 site link	Local international
Distributed Compute BookPrep	1 site cores * #sites	1 site link* #sites	Local+ international
Global BookPrep	1 site cores * #sites	1 site link* #sites	Local+ international

To further prove this, we have calculated these costs in more detail in terms of cost in dollars and time to complete processing. We used configuration parameters as presented in Table 5 below. We have based parameters on the publicly available Amazon costs as well as values we have measured from three Open Cirrus sites (see Table 2 and Table 3 earlier in the paper).

Table 5. Parameters used for Cost Analysis

External Providers Data		**Customer Demand**	
Compute small (per h)	$0.085	Location	# books
Storage (/GB/month)	$0.10	US	100
Data Transfer (/GB)	$0.12	Europe	100
Put Requests (/1,000)	$0.01	Asia	100
Get Requests (/ 10K)	$0.01	**Experiment-Specific Data**	
Ship local	$1	# pages	200
Ship International	$2.5	Process (/book/h)	4.5
Process (/page/min)	1.35	Raw book size (GB)	0.5
Upload time (MB/s)	1.45	book size (GB)	0.33
Ship duration local	5	Caching (#months)	24
Ship duration intn'l	10	# of nodes per site	10

For these parameters, we have calculated the cost and time to complete book processing and shipping and presented these values in Table 6 and Table 7 below.

Table 6. Cost Comparison of BookPrep Implementations

Configuration	Compute	Store	Network	Ship	Total
Single Site	$114.75	$204.00	$30.00	$600	$948.75
Distributed	$114.75	$204.00	$42.00	$600	$960.75
Global	$114.75	$204.00	$30.00	$300	$648.75

Table 7. Time Comparison of BookPrep Implementations

Configuration	Comp	Network	Ship	Total
Single Site	5.63	1.25	6.88	16.88
Distributed	1.88	0.58	2.46	12.46
Global	1.88	0.42	2.29	7.29

We have proven and quantified our hypothesis that globally distributed BookPrep can improve scalability and throughput by 31% and 32% respectively and the cost of operation by 56% when compared to single-site implementation.

VI. Lessons Learned

While doing this experiment we came to the following conclusions:

- Careful distribution is very important to match the requirements. Initially, the storage was kept centralized to reduce the costs, but subsequently when scalability became limiting factor, further distribution took place.
- Automation is important for deployment and for inferring the performance of individual sites. In retrospect, we should have created specs (application requirements) to explain to participating network sites how to get onto the BookPrep network.
- Heterogeneity of some of the systems prevented us from larger deployment (having better specs would help to recognize mismatches early in the process).
- A specific example of obstacle is the staged access to systems (from the staging server into the internal network) that required additional functionality in our deployment. VPNs are another case.
- Once initial setup was completed, it was very easy to do the deployment, which was a positive experience. This gave us confidence that it would be easy to scale this architecture to multiple regions and more machines within a single region.
- We experimented with different storage approaches; file servers, network file systems (NFS), and Cloud storage (S3 equivalent). The file servers and the network file servers were not able to handle the load; as the number of servers went up, we saw consistent failures. Reads were fine, but for writes, only reliable solution was S3 equivalent. This came at the expense of more bandwidth, latency and processing time.

VII. Future Work

We can schedule processing of larger sized books on machines that have larger memory and faster CPU. This will improve the performance. This could be done in a single install or distributed installs. However we may find that some installations support smaller configuration machines and others support larger configurations. We can compute the average processing time for different sizes and figure out if the performance can be improved by intelligently allocated books to the right kind of machines.

We will conduct more measurements and also do some more simulation beyond a simplified use case with 300 book requests. In particular we will emphasize the benefits of either configuration. We also plan to experiment with different applications to leverage this architecture.

Distributed meta-data management is another area of future work. If the number of books goes to millions, the scale will become an issue. Currently the meta-data is stored on the portal site in the SQL database, but we will need to move to a non-SQL data base eventually. Replication and caching of data to support efficient online viewing and publishing is another future item. Right now, data is being pushed on demand to where it will be printed, but we can improve on that.

Distributing process queue, the last remaining centralized component would improve reliability and business continuity through distribution. We would also like other developers to contribute modules for, e.g. pushing to different print service providers and image processing algorithms and different form factors (not just PDF).

VIII. Related Work

To our knowledge there are few examples of web services and applications that are dynamically partitioned across different and independent cloud platforms (i.e. cloud independent services). SmugMug [4], a photo web service, has used Amazon's EC-2, S3 combination to support dynamic scaling. Their solution is specifically targeted towards AWS. Other services do Cloud bursting between private and public Cloud. Other examples of services include Polyphony (a modular workflow orchestration), Scribd (conversion of PDFs, MS Word, and other documents into Web documents), Yelp (consumer review Web site), 3scale (SaaS infrastructure for document and content management), and ActualAnalytics (automated and assisted video content analysis).

However, most scalable services do run across replicated and homogeneous sites. Examples include Google [5], Yahoo [6], Amazon [3], eBay, etc. Rollout of services is also typical for most Web services. Our innovation is in pulling the new version from the repositories periodically. Our work has followed principles of Cloud platforms [7] and leveraged most of the benefits of Cloud Computing [8]0.

IX. Summary

We have presented a globally distributed implementation of BookPrep service that takes raw copies of books and converts them into printable versions. Our implementation currently executes on three open Cirrus sites, in Palo Alto, Georgia tech, and Karlsruhe institute of Technology.

As part of porting BookPrep we also evolved its architecture and configuration towards global deployment. We started from the original single site configuration, through distributed implementation that replicates compute part (both part of our previous work), and concluded with the current implementation that replicates all components except for the portal.

We have conducted a number of measurements for each site and used these measurements to do a more extensive performance comparison in terms of cost and time-to-execute. We have confirmed our intuition that globally distributed BookPrep substantially (>22%) reduces cost in our use case (primarily due to local shipping) and also reduces time-to-execute (41% and 56% respectively).

In the process of porting BookPrep we have confirmed the need for automating access to Open Cirrus sites, as non-negligible work had to be repeated for new sites.

There were a few surprises that we encountered while evolving BookPrep architecture. A) The first one was the ease with which porting BookPrep to other Open Cirrus sites took place (even though automation would have helped). B) Straightforward transition to VMs, earlier we used exclusively physical machines. C) The evolution of architecture, and in particular how suitable Cloud was for BookPrep with special relevance to compute and storage.

Acknowledgements

We are indebted to a number of researchers and developers who have helped us in the development of each of the systems. In particular, we would like to thank Karsten Schwan and Char Huneycutt from Georgia Tech, Marcel Kunze and Ahmad Hammad from KIT. Even though we did not initially used nodes from the following institutions we are thankful to Luke Jing Yuan from MIMOS, Han Namgoong and O.K. Min from ETRI, and Greg Ganger and Michael Stroucken from CMU.

References

[1] Prakash Reddy, Jian Fan, Steven Rosenberg, Andrew Bolwell, Jim Rowson. A web service for long tail book publishing. http://research.microsoft.com/en-us/um/cambridge/events/booksonline08/papers/p45.pdf.

[2] Prakash Reddy, James Rowson, Benedict Rozario, Shariff Dudekula, Anil Dev V., "A Print On Demand web service based on cloud computing," Proceedings of the 16th IEEE International Conference on High Performance Computing (HiPC), Dec 2009.

[3] Amazon Web Services: http://aws.amazon.com/ec2/.

[4] Cloud computing: The SmugMug Approach to using Amazon's EC2 and S3: http://opensource.sys-con.com/node/590285

[5] Sanjay Ghemawat et,al : The Google File System

[6] Hadoop:Tutotial. http://hadoop.apache.org/common/docs/r0.17.2/mapred_tutorial.html

[7] David Chappell : A short introduction to cloud platforms http://www.davidchappell.com/CloudPlatforms--Chappell.pdf

[8] Michael Armbrust, et al : Above the Clouds: A Berkeley View of Cloud Computing

[9] Avetisyan, A., Campbell, R., Gupta, I., Heath, M., Ko, S., Ganger, G., Kozuch, M., O'Hallaron, D., Kunze, M., Kwan, T., Lai, K., Lyons, M., Milojicic, D., Lee, H.Y., Soh, Ming., N.K., Luke, J.Y., Namgong, H., "Open Cirrus A Global Cloud Computing Testbed," IEEE Computer, vol 43, no 4, pp 42-50, April 2010