# BITS Pilani presentation

D. Powar
Lecturer,
BITS-Pilani, Hyderabad Campus

**BITS** Pilani
Hyderabad Campus

innovate achieve lead

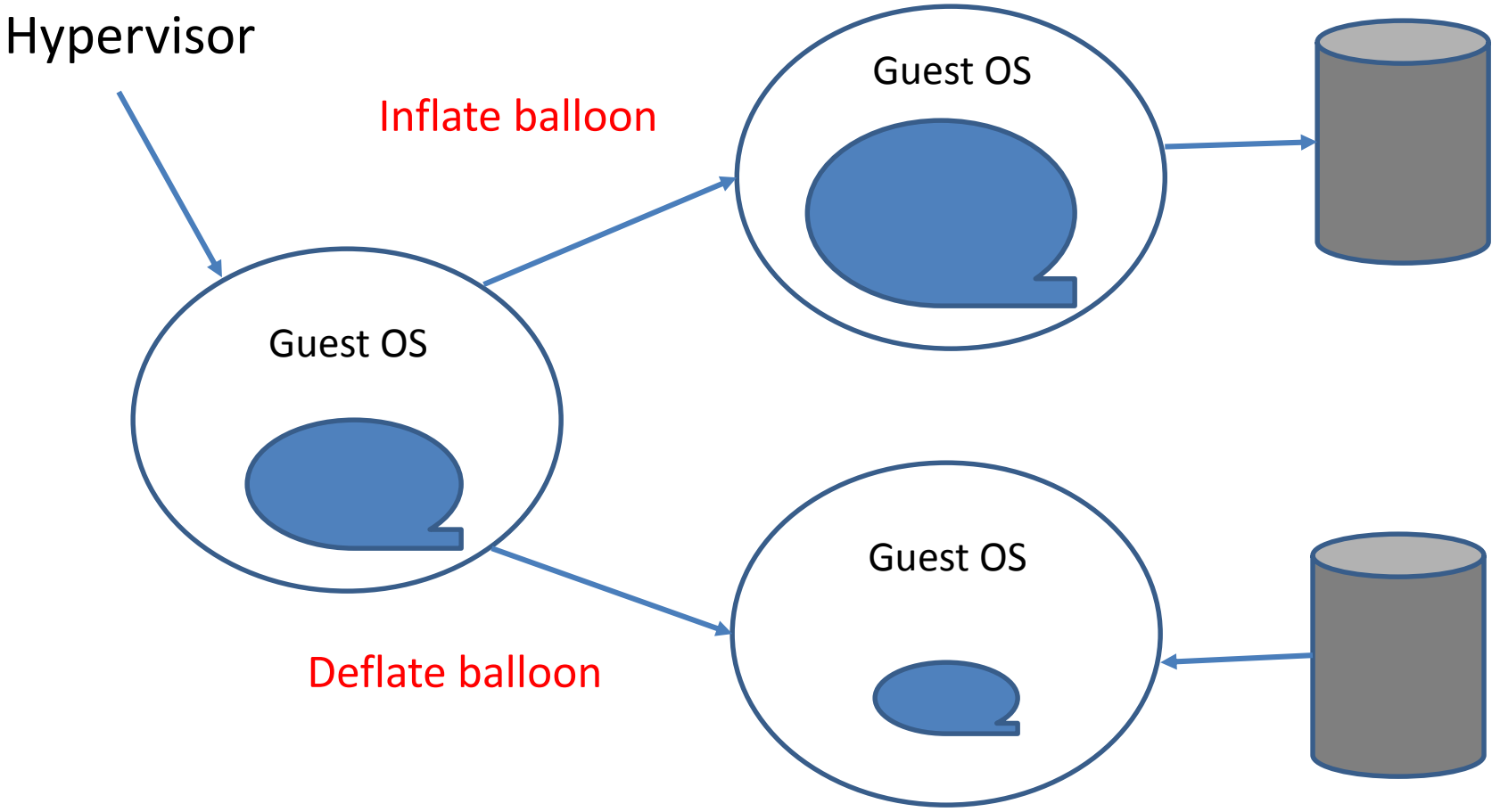
BITS Pilani
Hyderabad Campus

# CS ZG527
# Cloud Computing
# Lecture -04

# Objectives

- Ballooning in virtualization
- Introduction to SaaS
- Pros and Cons of SaaS model
- Applications of SaaS
- Traditional packaged Software Vs SaaS
- Examples of SaaS
- Case study

# Ballooning


Hypervisor
Inflate balloon
Guest OS
Guest OS
Guest OS
Deflate balloon

# SaaS (Software as a Service)

## - *No Worries - It's a Service*

# SaaS - Definition

- The most complete cloud computing service model is one in which the computing hardware and software, as well as the solution itself, are provided by a vendor as a complete service offering
- SaaS is a model where an application is hosted on a remote data center and provided as a service to customers across the internet
- In this model the provider takes care of all software development, maintenance and upgrades
- Salesforce.com is a common and popular example of a CRM SaaS application

- Customers use the software application using a browser.


http://cloudcomputingwire.com

# Is it customizable?

- Many people believe that SaaS software is not customizable, and in many SaaS applications this is indeed the case
  - user-centric application like office suite
- Many other SaaS solutions expose Application Programming Interfaces (API) to developers to allow them to create custom composite applications
  - Salesforce.com, Quicken.com, etc

- So, SaaS does not necessarily mean that the software is static or monolithic

# SaaS characteristics

- ✓ The software is available over the Internet globally through a browser on demand
- ✓ The typical license is subscription-based or usage-based and is billed on a recurring basis
- ✓ The software and the service are monitored and maintained by the vendor, regardless of where all the different software components are running
- ✓ Reduced distribution and maintenance costs and minimal end-user system costs generally make SaaS applications cheaper to use than their shrink-wrapped versions

# SaaS characteristics (contd..)

- ✓ Such applications feature automated upgrades, updates, and patch management and much faster rollout of changes
- ✓ SaaS applications often have a much lower barrier to entry than their locally installed competitors, a known recurring cost, and they scale on demand
- ✓ All users have the same version of the software, so each user's software is compatible with another's
- ✓ SaaS supports multiple users and provides a shared data model through a single-instance, multi-tenancy model

# SaaS - Pros

- No large upfront costs - usually free trials
- Anywhere, anytime, anyone - mobility
- Stay focused on business processes
- Change software to an Operating Expense instead of a Capital Purchase, making better accounting and budgeting sense.
- Create a consistent application environment for all users
- No concerns for cross platform support
- Easy Access
- Reduced piracy of your software
- Lower Cost
  - For an affordable monthly subscription
  - Implementation fees are significantly lower
- Continuous Technology Enhancements

# SaaS - Cons

- Initial time needed for licensing and agreements
  - *Trust*, or the lack thereof, is the number one factor blocking the adoption of software as a service (SaaS).
  - Centralized control
  - Possible erosion of customer privacy
- Absence of disconnected use
- Not suited to high volume data entry
- Broadband risk

# SaaS ??



- Imagine a system
  - where you don't have to buy new hardware or update software
  - where you pay nothing or pay as much as you use
  - where everything is done as a service: Infrastructure, computing, storage and usage
  - where you don't worry about your resources spent on Infrastructure security and operational security
  - where you cut your IT spending
  - where you have freedom of usage from anywhere with internet connectivity
  - which is eco-friendly

# Real world example:

**Traditional**

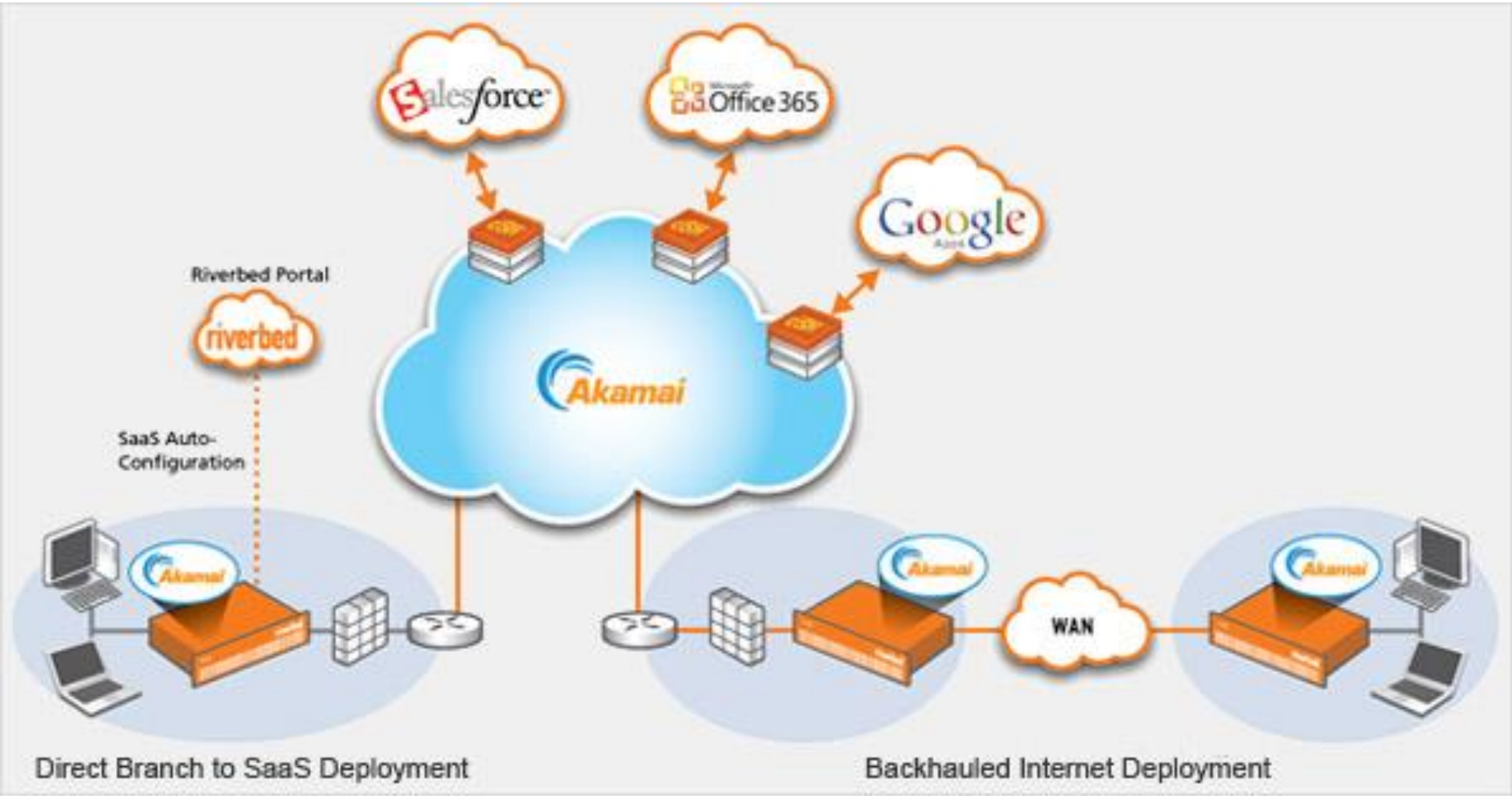
**Build Your Own**

**On-Demand Utility**

**Subscribe, Plug In, Use**

**Pay-per-Use**

# SaaS




Source: wiki

# Traditional packaged Software Vs SaaS



- Designed for customers to install, manage and maintain
- Architect solutions to be run by an individual company in a dedicated instantiation of the software
- Designed from the outset up for delivery as Internet-based services
- Designed to run thousands of different customers on a single code

# Traditional packaged Software Vs SaaS



- Infrequent, major upgrades every 18-24 months, sold individually to each installed base customer
- Version control
- Upgrade fee
- Fixing a problem for individual customer
- May use open APIs and Web services to facilitate integration, but each customer must typically pay for one-off integration work

- Frequent, "digestible" upgrades every 3-6 months to minimize customer disruption and enhance satisfaction
- Version control
- - do -
- Fixing a problem for one customer fixes it for everyone
- Streamlined, repeatable functionality via Web services, open APIs and standard connectors

# Example SaaS applications

- Salesforce.com

- Google Apps
  - ✓ Gmail, Google Groups, Google Calendar, Talk, Docs, etc
  - ✓ Google Apps Marketplace (Google apps for both free and for a fee)

- Microsoft Office 365
  - ✓ **Office 365** is a subscription-based online office and software plus services suite which offers access to various services and software built around the Microsoft Office platform

# SaaS examples



# Which applications are suitable?

- Any application can be deployed in this way. However communications over the Internet are not as fast as local connections - so ***leave any high volume*** data entry applications on your internal LAN or WAN. All the rest can go on the Internet under a SaaS approach

# Myths

- SaaS is still relatively new and untested
- SaaS is just another version of the failed ASP and hosting models of the past and will suffer the same fate as its predecessors
- SaaS only relieves companies of the upfront costs of traditional software licenses
- SaaS is only for small and mid-sized businesses and will not be accepted by large-scale organizations

# Myths (contd..)

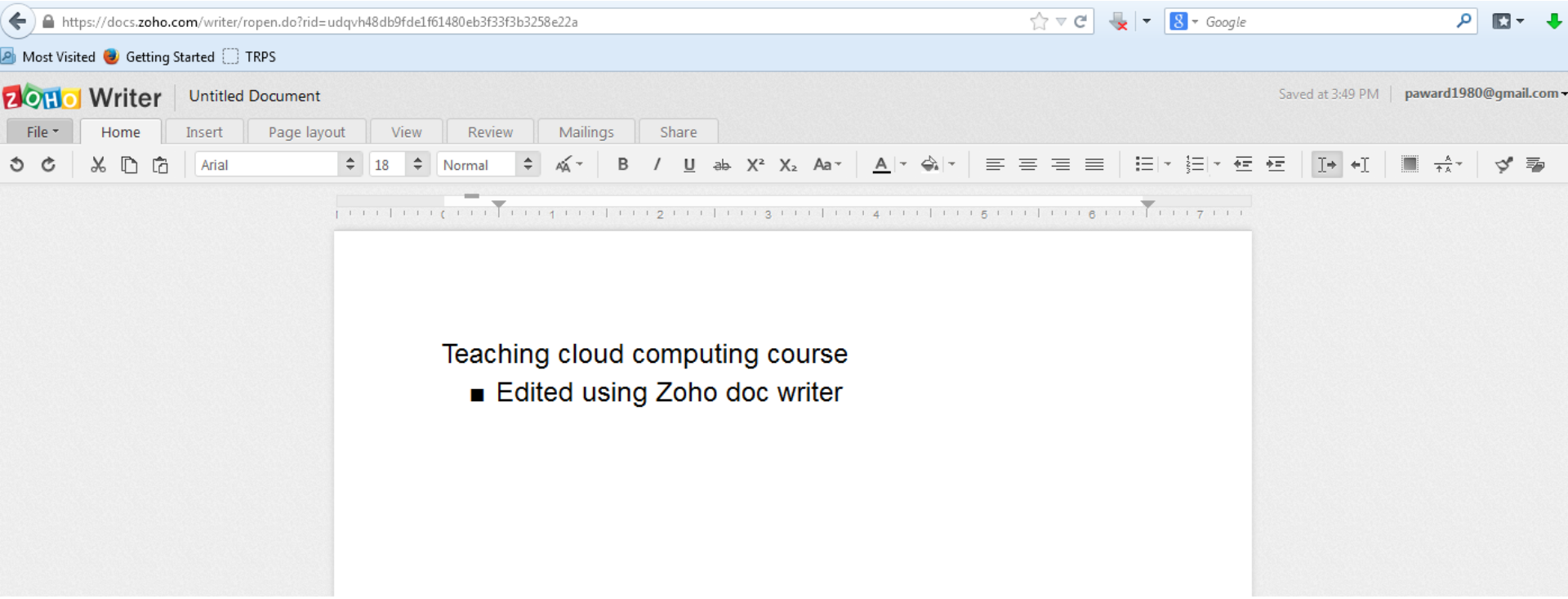
- SaaS only applies to applications such as CRM and Salesforce automation
- SaaS will only have a minor impact on the software industry and will fade over time
- It will be easy for the established software vendors to offer SaaS and dominate this market
- SaaS is only for corporate users

# Home work (Case studies)

1. Salesforce.com (customer relationship management (CRM)): whose solution offers sales, service, support, marketing, content, analytical analysis, and even collaboration through a platform called Chatter
2. Social computing (Facebook, Picasa, Twitter)

3. Quickbook - **QuickBooks** is an accounting software package
4. **GoToMeeting -** It is an online meeting, desktop sharing, and video conferencing software that enables the user to meet with other computer users, customers, clients or colleagues via the Internet in real time
5. Zoho Office Suite

# Zoho doc writer



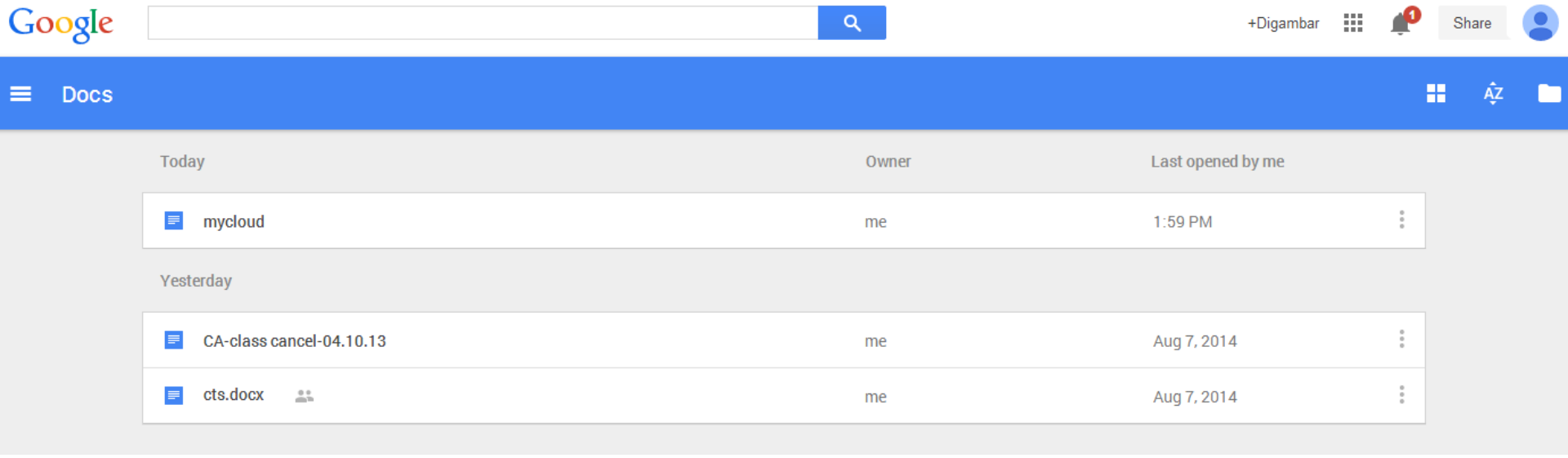
# Case study- DOCUMENT SERVICES: GOOGLE DOCS

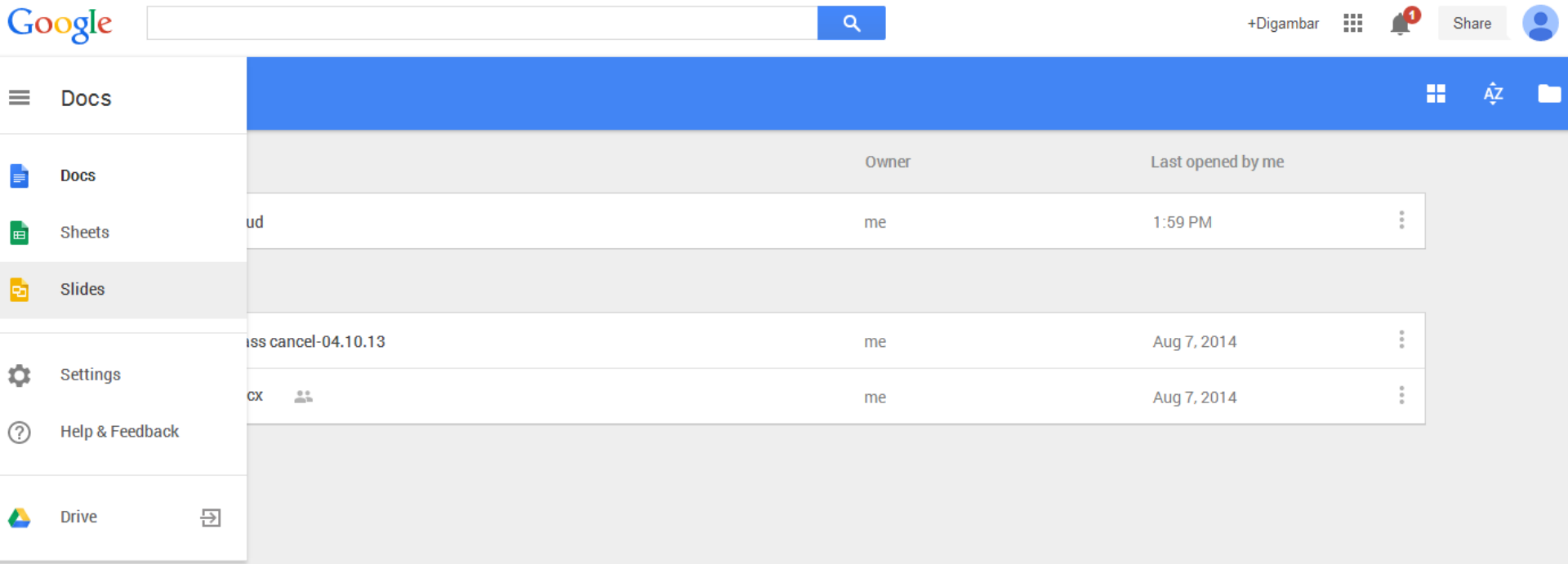


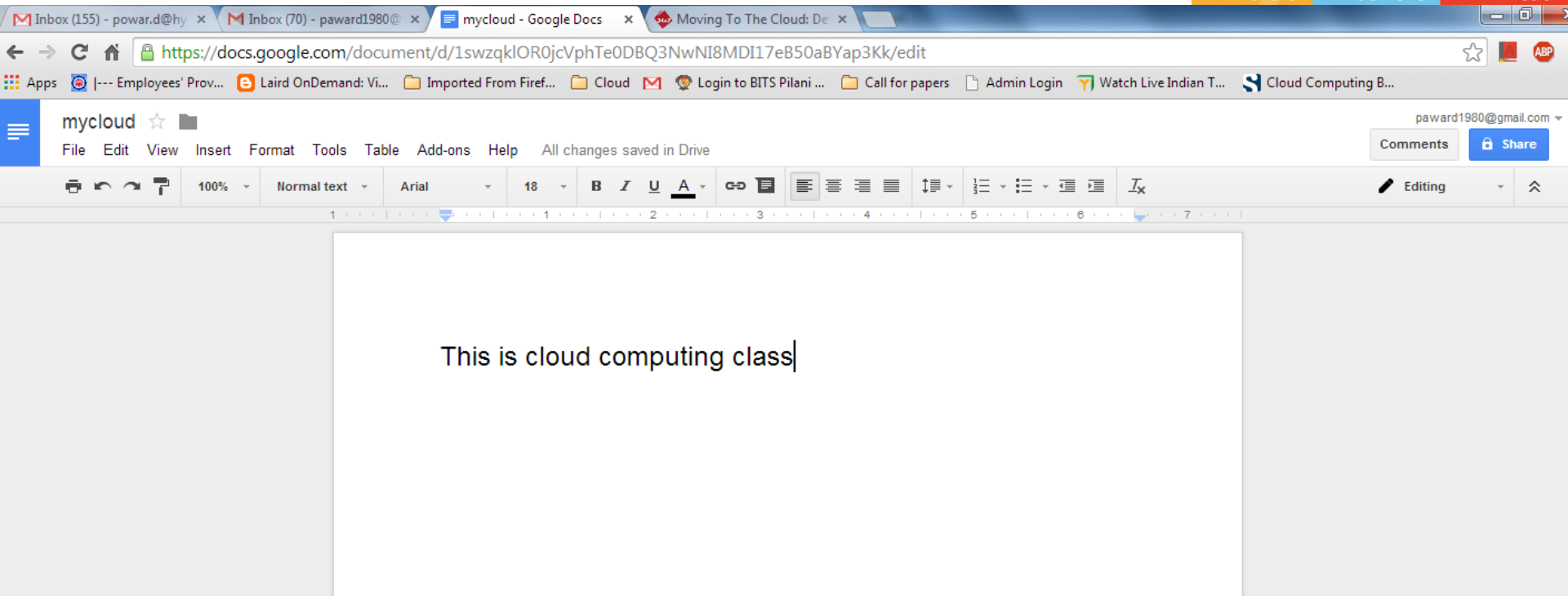
- Cloud computing SaaS service designed to enable uploading and sharing of documents as a persistent repository of information
- All the features of Google Docs can be accessed using a portal
- The developer APIs enable the usage of this cloud application from within other applications

- Many such cloud services also exists, www.dropbox.com, www.slideshare.com, www.scribed.com

# Google Docs Portal




Inbox (155) - powar.d@hy
Inbox (70) - paward1980@
mycloud - Google Docs
Moving To The Cloud: De
https://docs.google.com/document/d/1swzqklOR0jcVphTe0DBQ3NwNI8MDI17eB50aBYap3Kk/edit
Apps
|--- Employees' Prov...
Laird OnDemand: Vi...
Imported From Firef...
Cloud
Login to BITS Pilani ...
Call for papers
Admin Login
Watch Live Indian T...
Cloud Computing B...
mycloud
File Edit View Insert Format Tools Table Add-ons Help
All changes saved in Drive
paward1980@gmail.com
Comments
Share
100%
Normal text
Arial
18
Editing
This is cloud computing class

# Google Docs APIs



- Provides APIs that allow users to develop applications that upload documents to the Google Docs service and share documents.
- **Google Data Protocol** (**GDP**) provides a secure means for new applications to let end users access and update the data stored by many Google products.
- GDP uses GET and POST requests
- Users may also use the protocol directly using any of the supported programming languages provided by HTTP client libraries

# Example:

- Example application that demonstrates several features of document sharing <1. Book>
- The application first uploads a document onto Google Docs, and then shares the document to people on a mailing list (Google groups id), which also sends an email notifying those people about the document

- Google Docs packages that one needs to import are the following

```
import com.google.common.*;
import com.google.gdata.util.*;
import com.google.gdata.client.uploader.*;
import com.google.gdata.data.docs.*;
import com.google.gdata.data.media.*;
import com.google.gdata.data.acl.*;
```

- A snippet of Java code that uploads a file without taking care of upload errors is given below

## Java code that uploads a file to Google Docs

```
public DocumentListEntry uploadFile(String filepath, String title)
                    throws IOException, ServiceException,
                    DocumentListException {
        URL createUploadUrl = new URL
            ("https://docs.google.com/feeds/upload/default/private/
            full");
        DocsService service = new DocsService("Pustak Portal");
        service.setUserCredentials(gmail_user, gmail_pass);
        File myfile = new File(filepath);
        String mimeType = DocumentListEntry.MediaType.fromFileName(
                            file.getName()).getMimeType();

        DocumentEntry myDocument = new DocumentEntry();

        myDocument.setFile(myfile, mimeType);

        myDocument.setTitle(new PlainTextConstruct(title));
        return service.insert(createUploadUrl), myDocument);
}
```

Refer book for:

- Printing the details of the uploaded file
- Handling errors while uploading
- Sharing the document with a mailing list

# Embedding Google Docs in Other HTML Pages

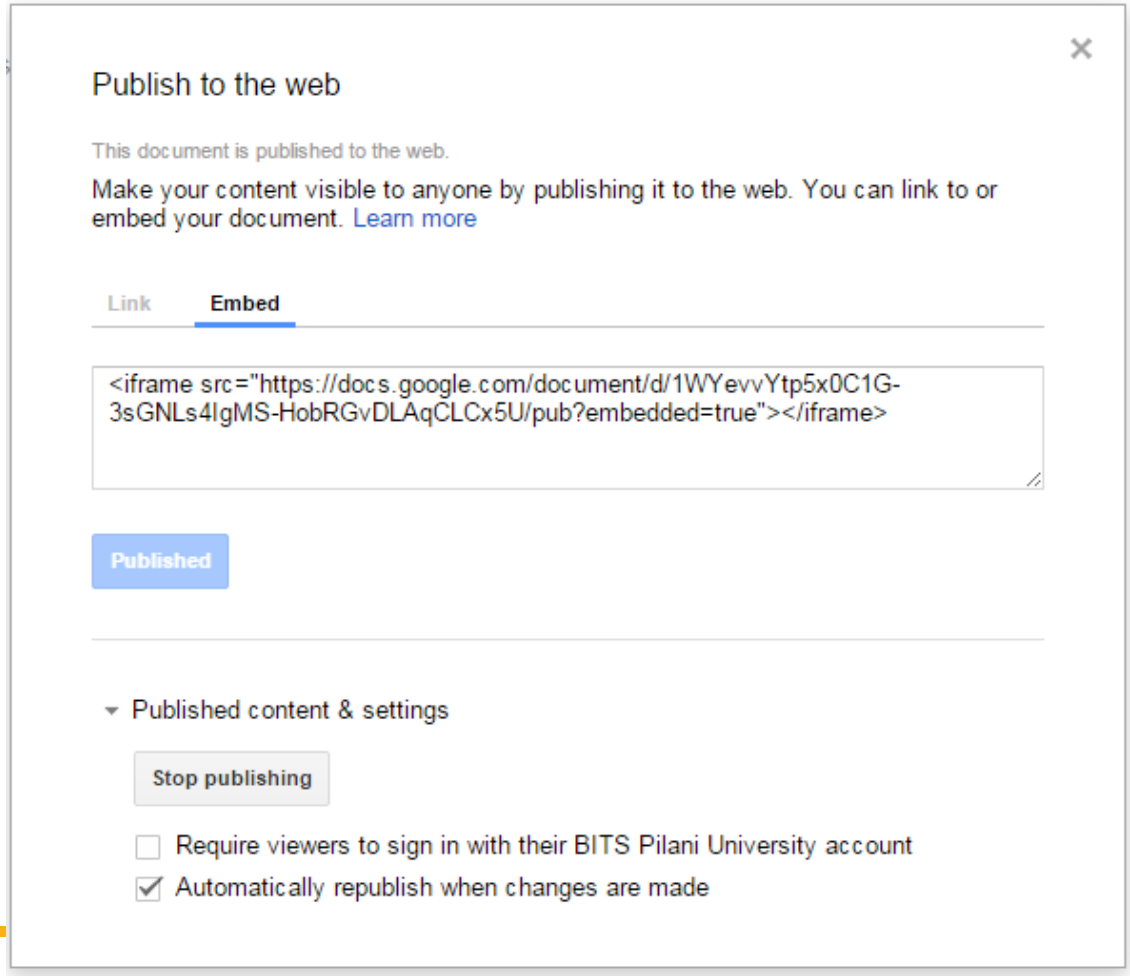
- Consider a scenario where you have your own web page and would like to embed Google Docs to use Google Docs as a back-end store.
  - Clicking on a link would display a document that is actually stored in Google Docs

## **Requirements**:

- Google doc API's for upload
- The unique URL of the document to be inserted is needed
- To get this unique URL, the file needs to be published as a web page

HTML code similar to the following, which has to be inserted into your web page:

```
<iframe
src="https://docs.google.com/document/d/1swzqklOR0jcVphTe0DBQ3NwNI8MDI17eB50aBYap3Kk/pub?embedded=true"></iframe>
```



# Summary:

- Introduction to SaaS
- Pros and Cons of SaaS model
- Applications of SaaS
- Traditional packaged Software Vs SaaS
- Examples of SaaS
- Case study