Condenser: Cloud Storage Gateway

Jesse Blum
for the
Horizon Digital Economy Research Institute
jesse.blum@nottingham.ac.uk

Nov. 2011

Version 0.2 draft

Revision History

Name	Date	Reason For Changes	Version
SRS-DLP-0.1	11-2011	First draft	0.1
SRS-DLP-0.2	11-2011	Research shift from logging to cloud gateway	0.2

Contents

List of Figures

1 Scope

This Software Requirements Specification (SRS) identifies the requirements for the Condenser Data Logging Platform.

1.1 Problem Statement

A trend across various Horizon projects (Energy and Relate for instance) is to provide distributed (immutable) sensor data capture, process and storage facilities. The connection between the local environment and external control and storage environment, however, is intermittent and unreliable. This disconnection results in many disadvantages such as the emergence of data silos and possible data loss. Horizon projects have heretofore provided bespoke solutions to the external connections, however these have become difficult to maintain with the increase of projects.

1.2 Objectives

The aim of the Condenser (meta)-project is to provide a configurable component for local logging of immutable data to a local server with intermittent synchronisation to external (possibly cloud-based) storage environments. The Condenser component will be made widely available and be data-type neutral.

1.3 Document Overview

Section 2 is divided into functional requirements and the non-functional requirements. The functional requirements explain the interactions between Condenser and its environment (including usage scenarios and cases). The non-functional requirements explain the quality constraints. Section 3 provides critical review of state-of-the-art existing options and existing components that may be leveraged in the delivery of Condenser.

2 Requirements

This section defines functional and non-functional requirements for the Condenser tool. It begins with an enumeration of usage scenarios that Condenser could be involved in. These are used to help determine the following requirements.

2.1 Actors

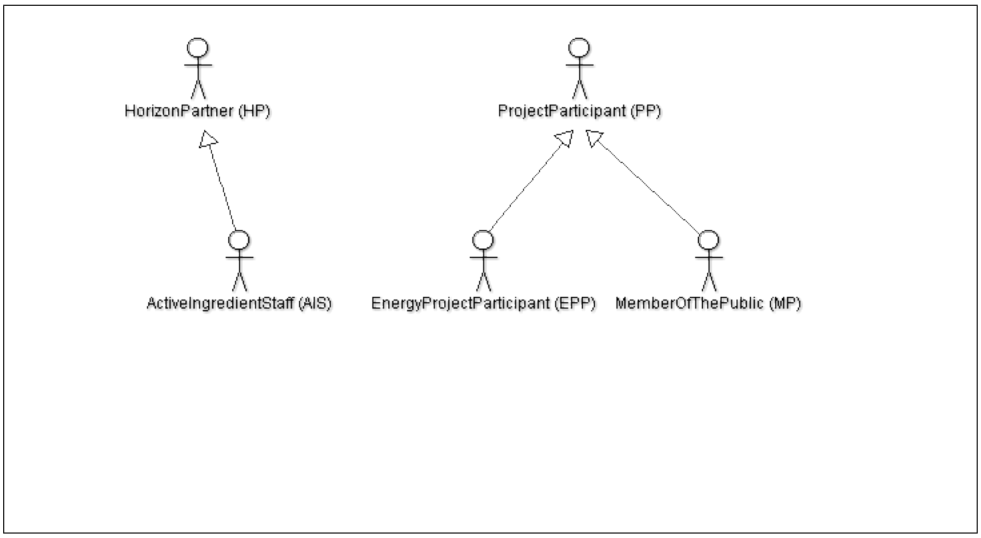

Figure 1: Actor relationships.

The actors shown in figure 1 are involved in the following usage scenarios and cases. They include:

- Horizon Project Staff (HPS) – These actors are involved in research projects as investigators.
- Horizon Partner (HP) – These actors are involved in research projects as investigators.
- Active Ingredient Staff (AIS) – These actors are a type of Horizon Partner.
- Project Participant (PP) – These actors are involved in research projects as participants.
- Member of the Public (MP) – These actors are a type of Project Participant recruited from the general public.
- Energy Project Participant (EPP) – These actors are a type of Project Participant recruited in particular for energy projects.

2.2 Usage Scenarios

2.2.1 Relate Project: Condenser Installation

Participating Actors: Anne:AIS , Harry:HPS
Event Flow:

1. Anne prepares to run an energy and climate data collection exercise in the Brazilian rain forest using open mobile sensing kits (OMSK).
2. Harry installs Condenser on the local server component of the OMSKs.
3. Harry configures a Cloud-server to accept data uploads from each OMSK.
4. Harry uses the Condenser RESTful interfaces to configure them to log the data from their local stores to the cloud when connectivity is available. He also configures them to run in low-powered mode by setting their reconnection attempts to a low amount and to only send data in aggregated bursts every hour.

2.2.2 Relate Project: Oasis of Connectivity

Overview: Condenser's behaviour in long period without connectivity
Participating Actors: Anne:AIS, Miguel:MP
Event Flow:

1. Anne conducts a public art activity in the Brazilian rain forest.
2. Anne gives Miguel an OMSK.
3. Miguel activates the OMSK as part of the activity.
4. On start up the OMSK activates Condenser and begins to capture temperature, humidity, C02 and energy information.
5. No network connectivity is available to Condenser so it evaluates the expected data load and determines that no action needs to occur until the next connection attempt. [Note that this would be much more complicated if Condenser determined that the current data capture rates exceeded the storage capacity]
6. Miguel returns the OMSK to Anne, and she returns to her office with it.
7. **Condenser** successfully attempts to connect to the network after its timeout.

8. **Condenser** transfers the data from the OMSK to the Cloud-store.

9. Anne looks at the day's data using the web-based visualisation tools. These have immediate access to the Cloud-data.

2.2.3 Energy Project: Trickle of Connectivity

Overview: Condenser's behaviour during intermittent connectivity
Participating Actors: Harry:HPS, Ernie:EPP
Event Flow:

1. Harry is conducting an energy study.

2. Harry sets up a Shiva Plug computer with an energy monitor.

3. Harry installs Condenser on the Shiva Plug.

4. Harry configures Condenser to attempt to reconnect whenever possible and to send discrete data points to a rack-mounted server using particular authentication details.

5. Harry installs the energy monitor in Ernie's home and configures it to use Ernie's home router.

6. **Condenser** attempts to transmit data as the energy monitor receives them, but connectivity is intermittent owing to Ernie's unstable Internet connection.

2.3 Use Cases

2.3.1 Installation

The following use cases pertain to installation of Condenser. Figure 2 shows a diagram depicting the relationships between the installation use cases.

Use Cases:

Install local server and DB
Participating Actors: Horizon project staff (HPS)
Event Flow:

1. HPS downloads (or compiles from source) the appropriate Condenser binaries from a well-known repository.

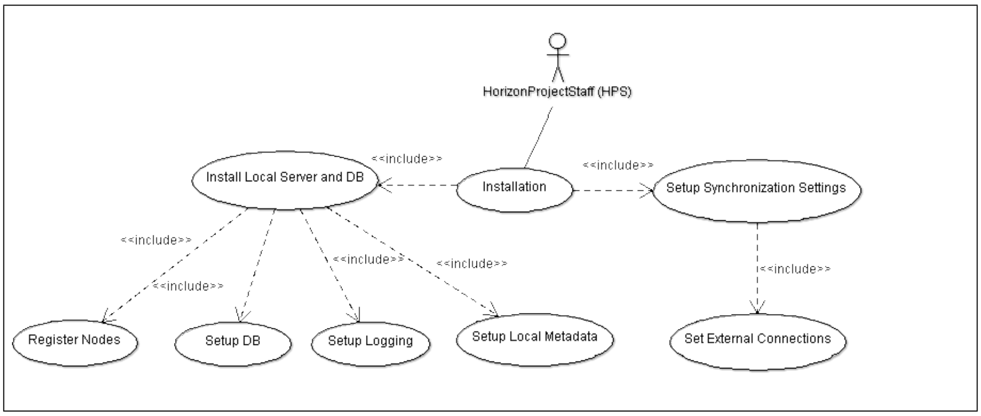

Figure 2: Use cases defining Condenser installation.

2. HPS sets up the Condenser file directory structure on the appliance.
3. HPS sets up Condenser's metadata (such as provenance information). See the setup local metadata use case.
4. HPS downloads and installs the database from a well-known repository (see the Setup DB use case).
5. HPS installs the Condenser service and configures it to cache data in the database.
6. HPS sets Condenser's connection settings (including any public key and password details).
7. HPS sets Condenser's logging settings (see the Setup Logging use case).

Entry Conditions:

- The given appliance meets the minimum hardware and operating system requirements.
- The given appliance is in a clean state (ie. there are no previous installation files/settings that can interfere with this installation).
- HPS has administrative control (and passwords) of the given appliance.
- HPS has Internet access.

Exit Conditions: Condenser is installed on the given appliance and ready to be externally synchronized.
Quality Requirements:

- Condenser is able to connect to the Internet and local sensor network.

- Data pushed to Condenser is cached appropriately.
- Condenser is able to push data upstream securely.

Set synchronization settings
Participating Actors: Horizon project staff (HPS)
Event Flow:

1. HPS accesses the synchronization settings web interface for the given device.
2. HPS sets connection settings to the external (possibly cloud) data storage system (see Set External Storage use case).
3. HPS sets synchronization rules (such as time-based or event-based synchronization).
4. HPS sets local data cache rules (how to handle limited data storage).

Entry Conditions:

- Condenser service is running.
- External (possibly cloud-based) storage has been setup and may be connected to remotely.
- HPS has Internet access.

Exit Conditions:

- Synchronization between Condenser's local cache and external data storage is up and running.
- Condenser's synchronization settings are viewable and update-able by an administrator with suitable privileges.

Quality Requirements:

- Connection settings (such as passwords/keys) to external storage are suitably encrypted.

Set external storage connection settings
Participating Actors: Horizon project staff (HPS)
Event Flow:

1. HPS accesses the external storage connection settings web interface for the given device.

2. HPS adds one or more connections to the connections store. This may be done by selecting from connection templates for well-known external storage or by adding in one's own bespoke connection settings (which can be stored for future use as a template).

3. HPS successfully tests the connection(s).

Entry Conditions: Includes Set Synchronization use case entry conditions.
Exit Conditions: Condenser is shown to be able to connect and transmit data to external storage.
Quality Requirements: Major external repositories should be supported through templates (such as Amazon S3, Windows SQL Azure, Rackspace CloudFile, Google Storage and traditional FTP).

Setup local database
Participating Actors: Horizon project staff (HPS)
Event Flow:

1. HPS downloads the database from a well-known repository.

2. HPS installs the database.

3. HPS sets up the database structure appropriate for the given project or application.

4. HPS configures limited data handling rules as appropriate for the given project. Such rules include how data, log files and metadata ought to be compressed, aggregated or removed in times of low local data storage space.

Entry Conditions: No other database is running on the given port.
Exit Conditions: The local database is setup and tested.

Setup logging
Participating Actors: Horizon project staff (HPS)
Event Flow:

1. HPS accesses the logging settings web interface for the given device.

2. HPS configures the logging settings.

3. HPS tests to make sure that activity is being logged appropriately.

Entry Conditions: Empty log settings and no log files present. **Exit Conditions:** Condenser is setup to log activity to a given degree of verbosity.
Quality Requirements:

- Different levels of logging are available.

- Options are avaialable to set the disk size for log storage.
- Options are avaialble for log cleanup rules.

Setup server metadata
Participating Actors: Horizon project staff (HPS)
Event Flow:

1. HPS accesses the metadata settings web interface for the given device.
2. HPS configures Condenser's provenance settings.
3. HPS adds any special metadata for the given project or application.

Entry Conditions: Empty metadata settings.
Exit Conditions: Condenser's metadata is setup.

Register nodes
Participating Actors: Horizon project staff (HPS)
Event Flow:

1. HPS accesses the node registry web interface for the given device.
2. HPS subscribes Condenser to one or more nodes (data publisher or broadcaster) to capture data for.
3. For each node HPS records data provenance information (such as node type – ie. type of sensor) and other metadata pertaining to the given node.
4. HPS ensures that each node can transmit data to Condenser and that the given data are stored locally.

Entry Conditions: No nodes are registered with Condenser.
Exit Conditions: one or more nodes is registered with Condenser.
Quality Requirements: It must be possible to quickly register a large number of nodes.

2.3.2 Local data Capture

The following use cases pertain to how Condenser records data local to it. Figure 3 shows a diagram depicting the relationships between the local data capture use cases.

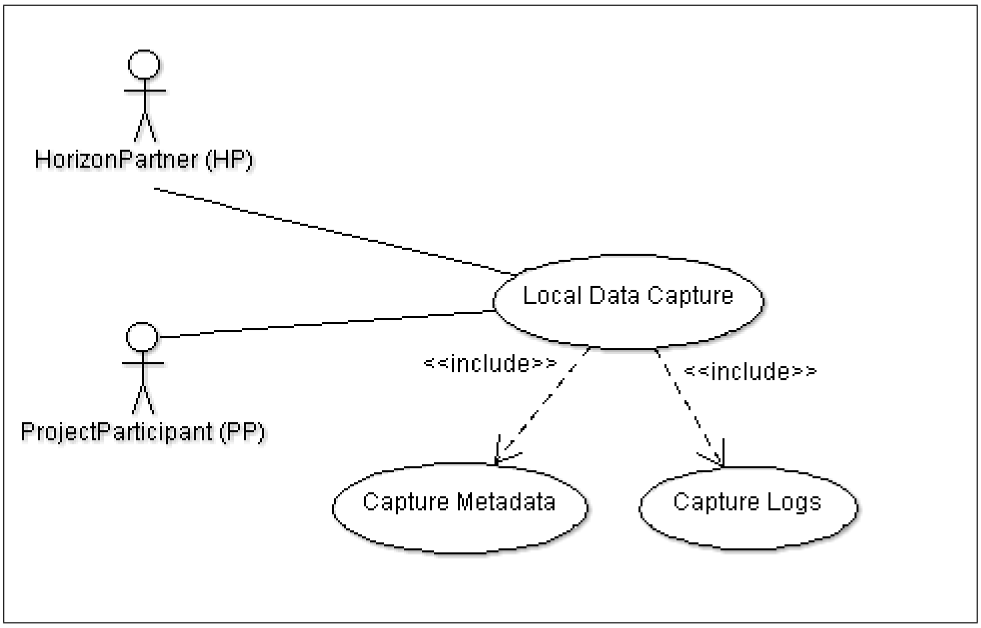

Figure 3: Use cases defining Condenser local data capture.

Use Cases:

Capture Data
Participating Actors: Horizon Partner (HP) and/or Project Participant(PP)
Event Flow:

1. The HP or PP initiates data transfer from one or more nodes to Condenser.
2. As Condenser receives data from its subscriptions, it stores the data and metadata (see Capture Metadata use case) to its database.
3. Condenser logs activity as per its logging settings (see Capture Logging Information use case) .

Entry Conditions: Condenser is installed in an environment and nodes are registered with it.
Exit Conditions: Condenser is recording (and possibly transferring) one or more data streams.
Quality Requirements:

- Stored data is time stamped and connected to appropriate metadata.
- Condenser should offer a high degree of reliability regarding data transfers to it (very little to no data loss)
- Condenser should be able to handle receiving secure (encrypted) and unsecured data streams.

Capture Metadata

Participating Actors: Horizon Partner (HP) and/or Project Participant(PP)
Event Flow:

1. Data streams begin to provide data points when the HP or PP initiate data transfer.

2. Each data point is captured in the database along with provenance metadata such as time stamps. Some of these metadata points will be links to descriptive records.

Entry Conditions: Metadata pertaining to given nodes was registered at time of node subscription.
Exit Conditions: Node data is described appropriately.
Quality Requirements: Metadata writing should not interfere with node data capture reliability.

Capture Logging Information

Participating Actors: Horizon Partner (HP) and/or Project Participant(PP)
Event Flow:

1. Data streams begin to provide data points when the HP or PP initiate data transfer.

2. Condenser should make log recordings of events such as error conditions within its own recording and/or transmission schemes, or anomalous node behaviour (such as if nodes go offline unexpectedly).

Entry Conditions: Logging is set up to at a sufficient level of verbosity.
Exit Conditions: Log records can be made of events.
Quality Requirements: Log writing should not interfere with node data capture reliability.

2.3.3 Data Transfer

The following use cases pertain to how Condenser transfers data to the cloud. Figure 4 shows a diagram depicting the relationships between the data transfer use cases.

Figure 4: Use cases defining Condenser data transfer.

Use Cases:

Data Transfer Service
Participating Actors: Horizon Partner (HP) and/or Project Participant(PP)
Event Flow:

1. The HP or PP initiates data transfer from one or more nodes to Condenser.
2. Data is cached locally.
3. When a data transfer event is initiated Condenser attempts to connect to and authenticate itself with one or more external data storage providers.
4. If there is external network connectivity then data is transferred off site (see time-orientated and event-orientated data transfer use cases) along with metadata see (see metadata transfer use case) and log information (see log file transfer use case). The data may be transferred periodically (see Time-orientated Data Transfer use case) or in response to events (see Event-orientated Data Transfer use case).
5. See Handle unavailable connectivity use case for when connectivity to the external network cannot be established.

Entry Conditions: Condenser is setup and has a stable Internet connection.
Exit Conditions: Data are transferred to external storage as well as cached locally.
Quality Requirements: Bandwidth used for data transfer should be minimised.

Time-orientated Data Transfer
Participating Actors: Horizon Partner (HP) and/or Project Participant(PP)
Event Flow:

1. The HP or PP initiates data transfer from one or more nodes to Condenser.
2. Condenser attempts to connect to the external data storage system at fixed intervals or at particular points in time as given in the configuration rules – see the set synchronization settings and set external storage connection settings use cases for more information.
3. If a connection is established, data, metadata and log files are transferred to external storage. These will be encrypted if the configuration settings are set for this.

4. If a connection fails the non-transferred data are enqueued for subsequent transfer attempts.

Entry Conditions: Condenser is configured to do time-based synchronization and has a stable Internet connection.
Exit Conditions: Data are transferred to external storage as well as cached locally.

Event-orientated Data Transfer
Participating Actors: Horizon Partner (HP) and/or Project Participant(PP)
Event Flow:

1. The HP or PP initiates data transfer from one or more nodes to Condenser.

2. Condenser attempts to connect to the external data storage system when particular events occur are per its configuration rules – see the set synchronization settings and set external storage connection settings use cases for more information. For instance, an event could be that the local data cache has reached a particular size or that Internet bandwidth is at a particular threshold.

3. If a connection is established, data, metadata and log files are transferred to external storage. These will be encrypted if the configuration settings are set for this.

4. If a connection fails the non-transferred data are enqueued for subsequent transfer attempts.

Entry Conditions: Condenser is configured to do event-based synchronization and has a stable Internet connection.
Exit Conditions: Data are transferred to external storage as well as cached locally.

Metadata Transfer
Participating Actors: Horizon Partner (HP) and/or Project Participant(PP)
Event Flow:

1. The HP or PP initiates data transfer from one or more nodes to Condenser.

2. If there is external network connectivity then metadata is transferred off site. Condenser should deduplicate metadata and only transfer that which has not already been transferred. Duplication could otherwise arise because some metadata will pertain to many data points.

Entry Conditions: Condenser is setup and has a stable Internet connection.
Exit Conditions: Metadata are transferred to external storage as well as cached locally.
Quality Requirements: Bandwidth used for metadata transfer should be minimised.

Log File Transfer
Participating Actors: Horizon Partner (HP) and/or Project Participant(PP)
Event Flow:

1. The HP or PP initiates data transfer from one or more nodes to Condenser.

2. If there is external network connectivity then log files are transferred off site. Condenser should use delta encoding to only transfer portions of log files that have not already been transferred.

Entry Conditions: Condenser is setup and has a stable Internet connection.
Exit Conditions: Log files are transferred to external storage as well as cached locally.
Quality Requirements: Bandwidth used for log file transfer should be minimised.

Handling limited local data storage
Participating Actors: Horizon Partner (HP) and/or Project Participant(PP)
Event Flow:

1. The HP or PP initiates data transfer from one or more nodes to Condenser.

2. If there is external network connectivity difficulties then its is possible that local data storage could fill up.

3. When the local data store is too full then Condenser should compress or aggregate or otherwise eliminate unnecessary data, metadata and log files as per Condenser's configuration settings (see Setup local database use case).

Entry Conditions: Condenser is configured with data handling rules.
Exit Conditions: Data are compressed, aggregated or deleted to maximise space for new data.

Handling limited external data storage
Participating Actors: Horizon Partner (HP) and/or Project Participant(PP)
Event Flow:

1. The HP or PP initiates data transfer from one or more nodes to Condenser.

2. If external storage capacity (or financial constraints) are exceeded then Condenser will act as if the network connection failed and cease data transfer.

3. Condenser will notify its administrator of the failure.

Entry Conditions: Condenser is configured with data handling rules.
Exit Conditions: Condenser suspends further data upload until it is unsuspended, and it notifies its administrator of the failure.

2.3.4 Local Service Review

The following use cases pertain to the review of service settings and logs of Condenser. Figure 5 shows a diagram depicting the relationships between the Local service review use cases.

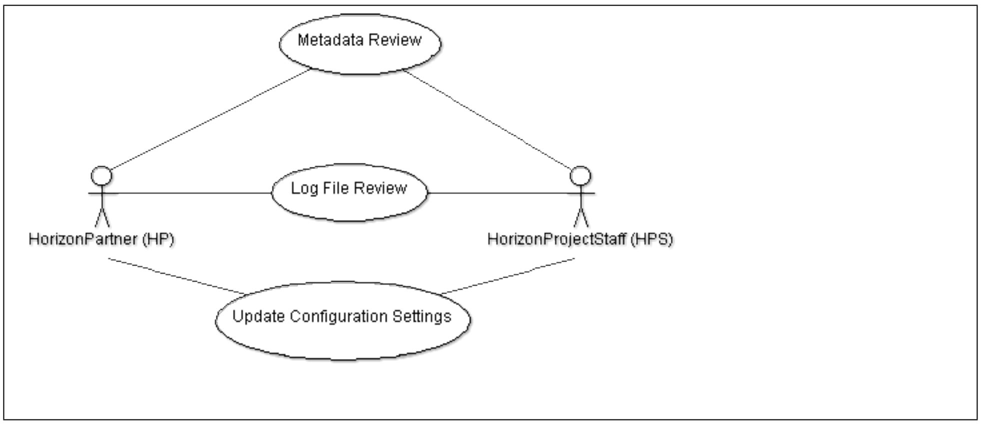

Figure 5: Use cases defining Condenser service review.

Use Cases:

Metadata Review
Participating Actors: Horizon Project Staff(HPS) and/or Horizon Partner (HP)
Event Flow:

1. HPS or HP accesses the metadata settings web interface for the given device.

2. HPS or HP re-configures Condenser's provenance settings.

3. HPS or HP adds any special metadata for the given project or application.

4. Condenser's new settings are saved as a new version. Old versions are archived for future inspection.

Entry Conditions: Condenser is running and can be connected to through the Internet.
Exit Conditions: Condenser has new provenance settings.
Quality Requirements: Metadata is maintained under version control.

Log Review
Participating Actors: Horizon Project Staff(HPS) and/or Horizon Partner (HP)
Event Flow:

1. HPS accesses the logging settings web interface for the given device.

2. HPS re-configures the logging settings.

3. HPS tests to make sure that activity is being logged appropriately.

Entry Conditions: Condenser is running and can be connected to through the Internet.
Exit Conditions: Condenser's logging settings are updated.

Update configuration settings
Participating Actors: Horizon Project Staff(HPS) and/or Horizon Partner (HP)
Event Flow:

1. HPS accesses the configuration settings web interface for the given device.

2. HPS re-configures the settings.

Entry Conditions: Condenser is running and can be connected to through the Internet.
Exit Conditions: Condenser's settings are updated.

2.3.5 Decommissioning

The following use cases pertain to the decomissioning of Condenser components. Figure 6 shows a diagram depicting the relationships between the decomissioning use cases.

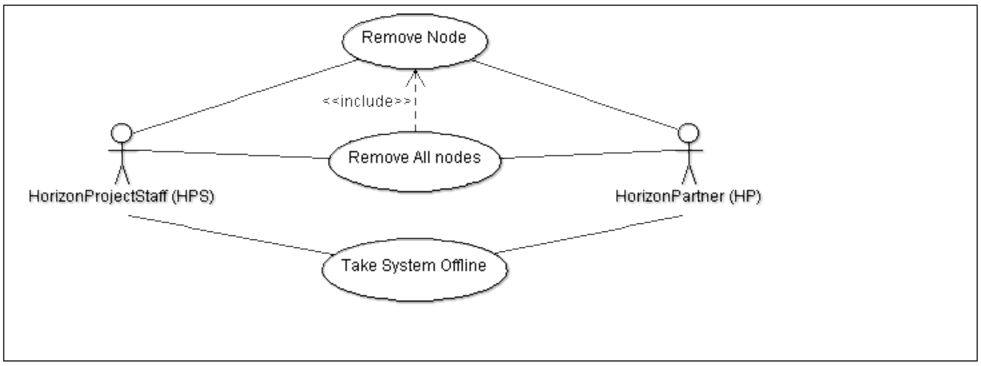

Figure 6: Use cases defining Condenser decomissioning.

Use Cases:

Removing a node
Participating Actors: Horizon Project Staff(HPS) and/or Horizon Partner (HP)
Event Flow:

1. HPS or HP accesses the node registry web interface for the given device.
2. HPS or HP un-subscribes to a node.
3. Condenser ceases to receive data from the node.

Entry Conditions: A node is registered with Condenser.
Exit Conditions: The node is no longer registered with Condenser.

Removing all nodes
Participating Actors: Horizon Project Staff(HPS) and/or Horizon Partner (HP)
Event Flow:

1. HPS or HP accesses the node registry web interface for the given device.
2. HPS or HP un-subscribes all nodes.

Entry Conditions: At least one node is registered with Condenser.
Exit Conditions: No nodes are registered with Condenser.

Taking Condenser offline
Participating Actors: Horizon Project Staff(HPS) and/or Horizon Partner (HP)
Event Flow:

1. HPS or HP accesses the uninstall web interface for the given device.
2. After a suitable warning HPS or HP initiate unistall.
3. Condenser attempts to backup all outstanding data, metadata and longs.
4. Condenser removes its local data storage contents and schema.
5. Condenser removes all of its installed software.

Entry Conditions: Condenser is running.
Exit Conditions:

1. All data cached by Condenser is replicated elsewhere.
2. Condenser is uninstalled leaving no footprint.

2.4 Functional Requirements

- Condenser is a cloud gateway infrastructural component for use in software projects requiring data transmission and intermittent synchronization with external environments.
- Functionality of Condenser is configurable through a RESTful interface. Clients can use HTTP calls to configure where and how often data is stored. Condenser can be used with cloud-based storage or more traditional server storage.
- Condenser is capable of handling disconnected operation by ensuring data are cached locally until network connection is resumed. Condenser evaluates expected data needs while offline and can be configured to clean, aggregate or delete data to handle limited local drive space.
- Condenser uses a web-based interface to set certain configuration settings.
- Condenser will automatically provide standard metadata.
- Condenser will log its own performance to an external repository (with configurable repository and verbosity).
- Condenser should run as a background service that turns on automatically at startup

2.5 Non-functional Requirements

2.5.1 Reliability and Security

Condenser should continue to work reliably during periods of network disconnect. Error conditions that arise during periods of disconnect should be logged in a similar fashion to sensor data and transmitted externally (to a configurable store) upon resumption of network connection. Configurable options should be made available to allow for secure data transmission and storage. Data store connection settings must be encrypted.

2.5.2 Test Data

Even though Condenser will be data-type neutral, it will be tested using temperature, humidity, energy and C02 sensors data. Data sets will include: time-series data, discreet values and aggregate information. Condenser will be tested with up to 2GB of data.

2.5.3 Performance

Condenser' s external storage performance during times of reliable network connection will be configurable is order for an administrator to choose the amount of bandwidth will be taken up transferring data offsite. Condenser should support the offsite data transfer of up to 4095 sources as well as one more for its own logging information.

When dealing with external storage, there is a bounded relationship between cost, capacity and latency. These factors need to be balanced for any Condenser solution.

2.5.4 Supportability

Condenser tests, code, installation, administration and usage will be well documented.

2.5.5 Implementation

The local logging Condenser component will be operational on a Plug computer. Off the shelf technology may be (and indeed is encouraged to be) used to get a working version of Condenser built as soon as possible. New developments should be test-driven with an emphasis on documentation.

2.5.6 Interface

Condenser should support a RESTful interface for its activation, configuration and data transmission. Condenser should be compatible with other Relate, HomeWork and Energy project components.

Condenser should be able to store data externally using a common solutions such as Amazon S3, Windows SQL Azure and FTP.

2.5.7 Operation

Condenser will be managed by Horizon staff or project clients.

2.5.8 Packaging

Condenser will be installed initially by Horizon project staff, with a view that client system administrators will be supported in the future. Condenser will initially be rolled-out for beta-testing in February, 2011.

2.5.9 Legal

Condenser will be licensed under The GNU Affero General Public License (AGPL 3). No liability will be assumed by Condenser's developers for losses incurred through its use since it is experimental and research driven software. Only Free (libre) software will be used for the development of Condenser.

Cloud Storage Synchronization Literature Search

Provenance JMB Nov. 2011 Scope: - Interested in determining current solutions for storing sensor data in the Cloud - required understanding of: * general cloud computing ideas and terminology * specific connections between Cloud Computing and Sensors * cloud (and general file/data) synchronization options * database replication * Weak n/w connection handling - During search expanded notion from simple synchronization at source to cloud gateway

Methods: Google scholar Wikipedia Paper references alternativeto ieee acm

Outcomes

Cloud Computing And Sensor Data Observations - Most resources are orientated at the Cloud-level (that is to tsay the back-end data centres) rather than the data input level - "Capacities continue to double each year, while access times are improving at 10 percent per year." - Jim Grey http://queue.acm.org/detail.cfm?id=864078 - The Big 4 (Amazon, Microsoft, Google and Rackspace) offer various storage models - According to http://www.broadbandspeedchecke 31.21 Mb/s download (337068 MB/d), 13.29 Mb/s upload (143532 MB/d) here at Horizon

General resources Above the Clouds: A Berkeley View of Cloud Computing - Definitions: * cloud = Datacentre h/w + s/w * SaaS = Services run on top of datacentre - Opportunities: * The cloud will become dominant because it is the most convenient way to store automated (large) datasets from sensors and users – services can then process, analyse and mashup the highly available data * infinite capacity on-demand - Obstacles: * Service availability * Data lock-in – possibly mitigated by API standardization * Data confidentiality – mitigated by encryption prior to storage * Data transfer bottlenecks – costs ($100-$150 / TB – unclear where figure comes from;) - SneakerNet is probably not viable here [Automated SneakerNet for smart buildings???]

Cloud Storage: Adoption, Practice and Deployment - Definitions: * cloud computing = WAN infrastructure * Cloud (with a capital C) = the public data network including the Internet - Provides a good overview of various issues such as security and data verification, performance, inter-cloud portability - Block Systems, File Systems (CIFS, NFS and GFS) and Object Storage

http://gigaom.com/2011/11/29/when-can-we-stop-talking-about-the-cloud/ “If I was explaining either service to my parents, for example, I would try not to use the term “cloud” at all. It wouldn”t really make any sense to them without an explanation, and once I started explaining it – how the cloud is a bunch of servers that Amazon or Google maintains in giant buildings that hold billions of individual files – it would actually make things worse. If I just pointed out that any files placed in a specific folder would automatically show up in other folders on different computers, then they would understand everything they need to know."

Cloud growth: http://gigaom.com/cloud/amazon-storing-more-than-449-billion-objects-in-s3/ http://gigaom.com/cloud/rackspace-cloud-revenue-keeps-rising-tops-100m/

Basic principles of Machine-to-Machine communication and its impact on telecommunications industry http://www.ericsson.com/res/docs/whitepapers/wp-50-billions.pdf

Sensor-Cloud Can We Plug Wireless Sensor Network to Cloud? - Raw data vs events simple or complex events - Proposed platform architecture (unclear where platform components live) [Of interest to this work (other components are downstream to sync data)] * WSN virtualisation manager - adapter provides raw byte stream from WSN to gateway which gateway enques as a raw packet in a buffer - Data processors affect enqueued packets - Command interpreter reprograms sensors via the gateway * Computation and storage manager - Processes and archives sensor data as XML A framework of sensor-cloud integration opportunities and challenges - Goal: connect sensors, people and s/w to build community-centric sensing applications - "current Cloud providers did not address the issue of integrating sensor network with Cloud applications and thus have no infrastructure to support this scenario" - Cloud vs grid (and how cloud is more appropriate for WSN data, but wsn-cloud integration is an unaddressed issue) - Proposed solution - Sensors stream data to gateway - Gateway sends data to pub-sub system in the cloud - Cloud services provide event monitoring and application services A design of flexible data channels for sensor-cloud integration - Attempts to integrate data channels from sensors to data analysis jobs in cloud - Their tools include Sensor Server - Runs on SN master node - activate/deactivate slave nodes - filter sn data - deliver data to clouds - Acts as a pull-based FTP server that makes sensors accessible to cloud services as files - Comms models - Push based * 1 data point = UDP packet or SNMP packet - Pull based WhereStore: location-based data storage for mobile devices interacting with the cloud - Filtered replication - keep data synced b/w set of peers - collections of items (each item containing data and metadata) - replicas of entire collections or subsets - filters (predicates over metadata) describe subsets - architecture - phone client, cloud server - WhereStore is stacked on top of: -filtered replication system - sync local cache on phone with cloud data items - smartphone replica contains items defined by its filter - subset of items for which totla capacity is below specified storage capacity - selection algorithm is given in paper - Based on Cimbiosys - Data accessed through callbacks -location service

A survey of mobile cloud computing - mobile cloud computing (MCC) - obstacles / MCC advantages performance battery life / Extending battery lifetime storage / Improving data storage capacity low bandwidth environment heterogeneity scalability / MCC apps can be scaled to meet the unpredictable user demands availability security reliability / Improving reliability (reduces the chance of data and application lost) privacy data integrity / [92] presents energy efficient solution authentication / [93] presents method for secure the data access suitable for mobile environments data access mechanism to manage, and synchronize data between mobile devices and the cloud to deal with changes of data on the cloud [102] addresses three main issues as follows: maintaining seamless connection between users and clouds, controlling cache consistency, and supporting data privacy

- Mobile Cloud Computing (MCC) architecture(Fig. 1) - Cloudlets: trusted, resource-rich computer or cluster of computers which is wellconnected to the Internet and available for use by nearby mobile devices Usable, lightweight and secure, architecture and programming interface for integration of Wireless Sensor Network to the Cloud. *** The following notes are direct copies of text: - The integration of wireless sensor networks, aggregated with sensor data but with limited processing power, with a resource-rich cloud computing infrastructure makes the integration beneficial in terms of i) scalability of data storage, ii) scalability of processing power for data mining and analysis, iii) access to the processing and storage infrastructure anywhere in the world, iv) resource optimization, v) and using pricing as one more criteria for the IT infrastructure. - Intortus, to enable this exploration of integrating sensor networks to the emerging cloud computing paradigm. - Intortus wherein sensor nodes collect data and send data to a gateway. The gateway aggregates data from the Wireless Sensor Network (WSN). - gateway in Intortus uses the same hardware as the sensor node (a wall powered mote) - web API with defined methods for accessing data - Intortus sends data from the gateway to the cloud securely using message authentication and encryption - user authentication data access policy and gateway authentication - Intortus currently, deals with a single application and is a framework dealing with solely data storage and communication to and fro the cloud in a secure manner - Intortus system consists of a cloud service that will be used to add data from the gateway to the cloud data store - Intortus provides the above first two services: 1. Store data from the WSNs securely. 2. Make this data available to users for viewing through web pages, to smart phone applications which will connect to Intortus and to other programs which will connect directly to Intortus 3. Get input from users via web pages, smart phone apps or other programs to control the applications in the WSNs. 4. Allow applications to be loaded from desktop computers through the cloud service to the WSN gateways, which will then push them onto individual motes in the WSN. - eMotes are loaded with software to collect, encode, and transmit data through wireless communication channels to the gateway - gateway is responsible to receive data from sensors and to dispatch it to be appropriate storage service hosted on cloud. The gateway structures data in the form of an XML and sends it over to the cloud. The data is sent to the cloud using web API method calls. The web API is created from a web service which is hosted in the cloud for interoperability from .NET Micro framework at the gateway to Java in the cloud. - data storage is Google App Engine - Intortus data storage API, used to store data from data aggregated at the gateway, creates a relational interface to the key value data structure so that the application providers need no knowledge of the Google App Engine data store structure while inserting data to the cloud. Data in the cloud is stored in an encrypted form for security purposes. Data can be accessed by an API which is a REST - Security in Intortus includes message authentication and encryption, data access, user authentication and gateway authentication - Challenges: * limited sensor node hardware resources make it hard to implement a security platform that incorporates the security requirements of Intortus. An autonomic cloud environment for hosting ECG data analysis services - Of little interest Cloud4Home– Enhancing Data Services with AtHome Clouds *** The following notes are direct copies of text (unless prefaced by ***): - current 'thin client' models in which end devices 'simply access the Internet' can suffer from high and variable delays in accessing and using remote resources - they are subject to challenges when devices must operate in disconnected - should leverage the lower costs of using local resources and exploiting locally available state, avoid potential issues with data privacy or security for cloud-based operation - should exploit Internet resources when those are not encumbered by undue costs like high latency or undue communication overheads - data services explored in the paper can tap into the aggregate resources offered by remote clouds, and they can leverage 'nearby' devices in home or office settings. The outcome is quality in service delivery that exceeds that of the pure 'in the cloud' or 'at the edge' service realizations. *** not really cloud is it – data centre oriented... - focus n services for storage, access, and manipulation of data for the home environment and when doing so, we leverage the VStore++ - virtualized object storage system that abstracts from an application where the objects it accesses are stored - VStore++ will track resource availability in order to direct requests to appropriate destinations based on their needs and/or resource availability, using a global indexing and monitoring infrastructure maintained during its operation - Cloud4Home comprised of dynamically varying sets of devices that cooperate to provide end users with seamless storage, access, and data manipulation services, including interactions with remote, publically available cloud platforms - vStore++ uses a standard file system to represent objects, using a one-to-one mapping of objects to files. In addition to object fetch and store operations, it supports an explicit process operation, which permits object manipulation functions to be associated with the object access *** overkill for WSN??? - metadata layer provides object lookup and transparent access to storage and services distributed across nodes in the home or remote clouds - metadata management layer is built as a distributed key-value store on top of a peer-to-peer overlay across all control domains in the home cloud - One or more nodes in the home cloud support a public cloud interface module, responsible for routing all remote cloud interactions Towards Reliable, Performant Workflows for Streaming-Applications on Cloud Platforms *** The following notes are direct copies of text (unless prefaced by ***): - propose and present a scientific workflow framework that supports streams as first-class data, and is optimized for performant and reliable execution across desktop and Cloud platforms - combine streaming data arriving from sensors with historic data available in file archives along with structured collections of weather forecast data that help the large scale computational model make an energy use prediction in near real time - workflow architecture that natively supports the three common data models found in science and engineering applications – files, structured collections and data streams – with the ability to seamlessly transition from one data model to another - logical stream abstractions have to be more robust than simple TCP sockets, given the unreliability and opaqueness introduced by operating in a distributed environment across desktop and Cloud with different characteristics from a typical local area network. Reliability of VMs hosting workflow tasks is another concern to be addressed. Too, there has to be intelligence to avoid costly (both in time and money) duplicate movement of the same logical stream - Streams are a continuous series of binary data - transient unless mapped to another data model - In the future, Cloud providers may provide such streams as IaaS abstraction - registry service that maintains a list of known streams and the endpoints where particular streams are provided - fault resistance: (1) transient or permanent loss of physical network, and (2) loss of virtual machines in the Cloud or services running on them.

Cloud experiences Adaptive Rate Stream Processing for Smart Grid Applications on Clouds - adaptive rate control to throttle the rate of generation of power events by smart meters, which meets accuracy requirements of smart grid applications while consuming 50% lesser bandwidth resources in the Cloud - power events generated at a peak rate of 1 KB/min from 1.4 M consumers in the Los Angeles Smart Grid will require 2 TB/day of streaming data to be processed and analyzed at an average cumulative bandwidth of 200Mbps - need for an adaptive stream rate control mechanism for generating power usage events - use the difference between the available power capacity at the utility and the current cumulative power usage by the consumers as our throttle function - disadvantages of static publishing rate: - setting too high a rate at which power events are published will cause excess resources (bandwidth, compute VMs for stream processing) to be utilized - setting too low a static rate at which power events are published can cause the utility to miss detecting a breach of a power usage threshold during peak load periods - adaptation logic (throttle) - as the total power usage within the utility approaches total available capacity, power usage events are required more frequently to detect/forecast a peak load event with low latency. - As future work, we plan to evaluate the scalability of the stream processing system with increasing number of VM instances, with the eventual goal of scaling up to 1.4 million smart meter streams that is expected in the Los Angeles Smart Grid. Both throughput and latency of the pipeline need to be measured as the stream rates adapt. Additional factors like availability of compute resources (# of VMs), throughput of the stream processing system, and cost trade-off between Cloud resource usage and power conserved can be incorporated into the throttle policy.

Cloud Storage Gateway General resources wikipedia.org A cloud storage gateway is a network appliance or server which resides at the customer premises and translates cloud storage APIs such as SOAP or REST to block-based storage protocols such as iSCSI or Fibre Channel or file-based interfaces such as NFS or CIFS.

CloudStorageSurvey–Avere - make cloud storage appear as local storage - translates cloud storage APIs to file interfaces or storage transport protocols - gateway resides at customer site - can be dedicated appliance/server, virtual appliance/server, or s/w application - features: compression, deduplication, encryption, caching, backup and recovery - Provides useful table columns for comparing cloud grteways, but unfortunately the table is incomplete (only the Avere FXT was examined and these cost $50000+)

http://gladinet.blogspot.com/2010/09/how-to-pick-cloud-storage-gateway.html - pick a cloud storage gateway, there are several questions: - 3 forms: s/w, virt. applicance, physical - which supported cloud storage providers - how to store data on cloud: - block level - file level - what encryption/compression algo used

http://www.cloudswitch.com/page/now-everybody-wants-a-cloud-gateway - gateway simplifies the integration and management of cloud resources so people can get on with using the cloud rather than struggling to make it work

http://www.cloudswitch.com/blog/category/Cloud- a well-designed cloud gateway needs to: - guarantee security - gateway should allow users and administrators to monitor and manage applications running in a cloud as if they were running locally, using existing tools and polices in a single, integrated environment - protect roles and access

www.nasuni.com/blog/28dirtysecrets5weaknessesofcloudstorage - gateway or device (virtual or physical) that needs to be installed at your site - beenfits: - improved performance - security - compression - dedup - WAN optimization - cache data - limitations - internet connection (can be mitigated usinga bulk data migration service) - Caches are not perfect and sometimes you will get read and write misses - write miss: the space you need to write to in the cache is temporarily full and has not been freed up by offloading/sending data to the cloud - ability to feed data to the appliance most likely is significantly faster than your Internet connection - You can't get at your data without the appliance because the appliance adds a level of security - cloud storage gateways change the data written to them -

compression, dedeup, encrypted, chunked for parallelism

http://www.nasuni.com/blog/32blocksvsfileswhichapproachisbetterforcloud - block-based gateway - gateway works directly with blocks, the 512-byte fundamental units of storage - raw chunks of storage on a drive - Applications assume that blocks exist in local, fast storage - cloud: connections between the applications and the storage can become strained (latency issues) - Intelligent caching is really, really difficult - Block-based gateways restore at the volume level, essentially restoring the blocks in a volume to a previous point in time - file-based gateway - Files provide a clear picture of the relationships among the blocks that support them - File-based gateways provide restore capabilities at the file, directory, or complete file system level - best for unstructured data Commercial offerings Atmos Ctera Nirvanix Citrix http://www.citrix.com/English/ps2/products/product.asp?contentID=2311977 http://www.opensourcerack.com/2011/05/25/citrix-netscaler-cloud-gateway-a-product-tour/ Gladinet Nasuni Cloudstore Open http://openstack.org/ http://docs.openstack.org/diablo/openstack-compute/starter/content/ch05s01s02.html Nimbus Project https://github.com/nimbusproject/nimbus Cumulus Cumulus: an open source storage cloud for science http://www.nimbusproject.org Eucalyptus http://www.cca08.org/papers/Paper32-Daniel-Nurmi.pdf

Synchronization Tools General resources Synch comparison chart File synch alternatives VCS discussion en.wikipedia.org/wiki/Comparisonoffilesynchronizationsoftware Cimbiosys: A platform for content-based partial replication

Common synch tools rsync tutorial algorithm examples DeltaCopy (Windows' rsync) Unison Open Source by Benjamin C. Pierce Mac, Windows, Linux http://www.stanford.edu/ pgbovine/unisonguide.htm http://www.stanford.edu/ pgbovine/uni for-your-mom.htm What's in Unison File Synchronization with Unison

Open source online tools SparkleShare https://github.com/hbons/SparkleShare http://sparkleshare.org/ http://www.makeuseof.com/tag/sparkleshare-great-open-source-alternative-dropbox-linux-mac/ dvcs-autosync http://gitorious.org/dvcs-autosync explanatory article http://www.mayrhofer.eu.org/dvcs-autosync git-annex

Online services symform http://www.symform.com/resilient-storage-architecture.aspx Windows only LiveMesh http://www.codeproject.com/Articles/37200/Cloud-Based-Source-Control-using-Live-Mesh-and-Git Syncany open-source cloud storage and filesharing application https://blueprints.launchpad.net/syncany/+spec/xmpp-notifications SpiderOak Free with limited functionality by Spideroak Mac, iPhone, iPad, Windows, Linux, Online, Android Dropbox Free with limited functionality Mac, iPhone, iPad, Windows, Linux, Online, Android, Blackberry Ubuntu One Free with limited functionality by Canonical Ltd. Windows, Linux, Online, Android SugarSync Free with limited functionality by SugarSync Mac, iPhone, iPad, Windows, Win Mobile, Online, Android, S60, Blackberry Wuala Free by LaCie Mac, iPhone, iPad, Windows, Linux, Online, Android CrashPlan Free with limited functionality by Code42 Mac, Windows, Linux, Online Box.net Free with limited functionality by Box.net iPhone, iPad, Windows, Online, Android, Blackberry Syncplicity Free with limited functionality by Syncplicity Mac, iPhone, Windows, Online Team-Drive Free by TeamDrive Systems GmbH Mac, Windows, Linux, Online ZumoDrive Free with limited functionality by Zecter Inc. Mac, iPhone, iPad, Windows, Linux, Online, Android Tonido Free by CodeLathe Mac, iPhone, iPad, Windows, Win Phone 7, Linux, Online, Android, Blackberry Minus Free by Minus Inc Mac, iPhone, iPad, Windows, Win Phone 7, Linux, Online, Android, Android Tablet, Blackberry Duplicati Open Source by Kenneth Skovhede, mortenmie, et al Windows, Linux Banckle File Sharing Free by Usman Sarfraz Mac, iPad, Windows, Online ownCloud Open Source by ownCloud team Acid Rain Open Source Mac, Windows, Linux, Online Sharebox lipsync diagram CPISynch On the scalability of data synchronization protocols for PDAs and mobile devices

Personal Cloud servers Pogoplug Iomega ix2-200 and ix4-200d network storage devices

(Distributed) File Systems General Resources Cloud-based synchronization of distributed file system hierarchies Systems HekaFS http://git.fedorahosted.org/git/?p=CloudF Coda GlusterFS RFS RFSa network file system for mobile devices and the cloud [102] WebDAV FUSE

Cloud storage services general resources Cloud storage survey (unpub.) An automated approach to cloud storage service selection MetaStoragecamera-ready Cloud Storage: Adoption, Practice and Deployment - Best practices * Chosing a provider: redundancy, fail-over, versioning, data back-up, mgmt console, pricing * local storage as well as cloud? Providers Amazon S3 Windows Azure ATandT Synaptic Storage Rackspace CloudFile Peer1 CloudOne Nirvanix Mezeo Google Storage traditional FTP WebDav server Pachube RESTful Environment, Datastream and Datapoint model Push and Pull capabilities with “live” and “frozen” status Supports HTTPS/SSL Authentication is handled using API keys.

Database Replication Unstructured CouchDB MongoDB

Weak connections weaklyconnectedusers Peer-to-peer Data Replication Meets Delay Tolerant Networking Deduplication Building a high-performance deduplication system

Cloud data security Challenges in Secure Sensor-Cloud Computing http://gigaom.com/cloud/the-cloud-meets-the-law-where-wikileaks-went-wrong/ Structured Encryption and Controlled Disclosure Computing Arbitrary Functions of Encrypted Data Every Cloud has An Encrypted Lining: The Effectiveness of Cryptography in Cloud Computing Fully homomorphic encryption using ideal lattices Distributing data for secure database services Addressing cloud computing security issues Silverline toward data confidentiality in storage-intensive cloud applications [92] Energy-efficient incremental integrity for securing storage in mobile cloud computing Lightweight and Compromise Resilient Storage Outsourcing with Distributed Secure Accessibility in Mobile Cloud Computing [93] Authentication in the clouds: a framework and its application to mobile users

3 References

References

[AFG+09] M. Armbrust, A. Fox, R. Griffith, A.D. Joseph, R.H. Katz, A. Konwinski, G. Lee, D.A. Patterson, A. Rabkin, I. Stoica, et al. Above the clouds: A berkeley view of cloud computing. *EECS Department, University of California, Berkeley, Tech. Rep. UCB/EECS-2009-28*, 1:1, 2009.

[AST02] S. Agarwal, D. Starobinski, and A. Trachtenberg. On the scalability of data synchronization protocols for pdas and mobile devices. *Network, IEEE*, 16(4):22–28, 2002.

[Bri09] S. Briefings. Can we plug wireless sensor network to cloud? *cloud computing: pinnacle of IT Infrastructure democratization*, 7:33, 2009.

[DZP+11] Y. Dong, H. Zhu, J. Peng, F. Wang, M.P. Mesnier, D. Wang, and S.C. Chan. Rfs: a network file system for mobile devices and the cloud. *ACM SIGOPS Operating Systems Review*, 45(1):101–111, 2011.

[HSH09] M.M. Hassan, B. Song, and E.N. Huh. A framework of sensor-cloud integration opportunities and challenges. In *Proceedings of the 3rd International Conference on Ubiquitous Information Management and Communication*, pages 618–626. ACM, 2009.

[LH10] K. Lee and D. Hughes. System architecture directions for tangible cloud computing. In *2010 First ACIS International Symposium on Cryptography, and Network Security, Data Mining and Knowledge Discovery, E-Commerce and Its Applications, and Embedded Systems*, pages 258–262. IEEE, 2010.

[MF11] J. Melchor and M. Fukuda. A design of flexible data channels for sensor-cloud integration. In *Systems Engineering (ICSEng), 2011 21st International Conference on*, pages 251–256. IEEE, 2011.

[Pac11] Pachube. *Pachube API Documentation*, 2011.

[Pat11] S.K. Patil. *Usable, lightweight and secure, architecture and programming interface for integration of Wireless Sensor Network to the Cloud.* PhD thesis, The Ohio State University, 2011.

[SMT10] P. Stuedi, I. Mohomed, and D. Terry. Wherestore: location-based data storage for mobile devices interacting with the cloud.

In *Proceedings of the 1st ACM Workshop on Mobile Cloud Computing & Services: Social Networks and Beyond*, page 1. ACM, 2010.

[UFB10] S. Uppoor, M.D. Flouris, and A. Bilas. Cloud-based synchronization of distributed file system hierarchies. In *Cluster Computing Workshops and Posters (CLUSTER WORKSHOPS), 2010 IEEE International Conference on*, pages 1–4. IEEE, 2010.