

# Odyssey

---

## Agile Application-Aware Adaptation for Mobility

Presented by  
Sumeet Singh

# Overview

---

- Motivations
- Key Observations
- Goals & Requirements
- Application-Aware Adaptation (Model)
- Design & Implementation
- API
- Example Applications
- Evaluation

# Motivations

---

- Mobile Hosts have limited resources
  - Bandwidth
  - CPU cycles
  - Battery power
  - ...
- Concurrently running applications contend for the same resources
- Unpredictable variation in network quality

# Key Observations

---

- Automatically adapt to changes in available resources by:
  - a) Inducing applications to vary their resource requirements
  - b) Centralize resource management to enhance performance of concurrent applications
- How ? Make the OS & the app share information:
  - Let the application stipulate to the OS it's resource requirements
  - Let the OS provide back to the client regularly updated information on availability of resources

# Goals & Requirements

---

- **Fidelity**
  - the degree to which data presented at client matches the reference copy at the server
- **Concurrency**
  - ability to execute multiple independent applications on a mobile client concurrently
- **Agility**
  - speed and accuracy with which changes are detected and acted upon

# Fidelity

---

- The degree to which data presented at client matches the reference copy at the server
  - Fidelity has many dimensions
  - Dimensions of fidelity are dependent on the data in question
  - Adaptation depends on both the type of data as well as the tradeoffs the application may choose to make.

# Concurrency

---

- ability to execute multiple independent applications on a mobile client concurrently
  - Coordinate resource management across applications vs. each application assuming that it has full use of available resource (ex network bandwidth).

# Agility

---

- Speed and accuracy with which changes are detected and acted upon
- Why is agility important?
  - Changes are large and erratic in mobile environment
- Why changes occur?
  - Variation in supply of resources
  - Variation in demand by concurrent applications
- Sensitivity to changes in different resources
  - Change in net bandwidth v change in battery power

# Application-Aware Adaptation

---

- The system monitors resource levels and notifies applications of relevant changes
- Each application independently decides how best to adapt when notified

# Application-Aware Adaptation

---

- Application diversity and concurrency
  - Diversity by allowing applications to determine the mapping of resource levels to fidelity levels
  - Concurrency by allowing the system to retain control of resource monitoring

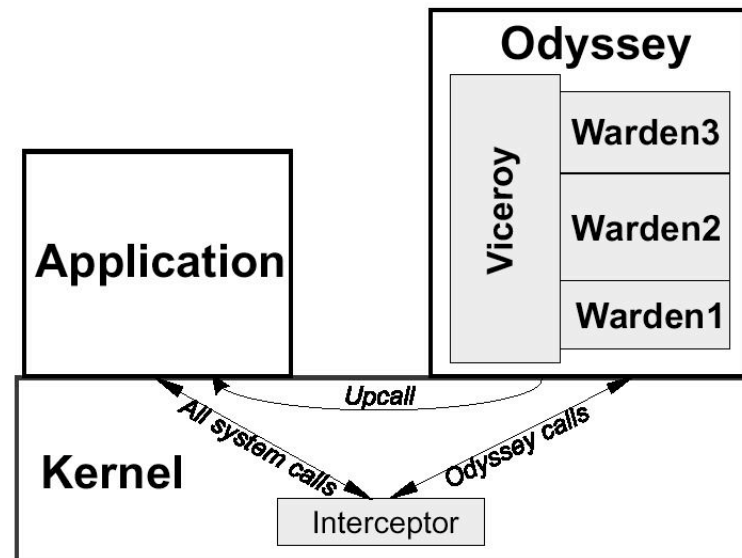
# Application-Aware Adaptation

---

- Type awareness incorporated into the system for efficient resource usage.
  - Wide disparity in properties of various data types
  - Impossible to optimize without some system level knowledge of type
  - Decisions like, which compression algorithm to use? Which transport protocol to use? etc.

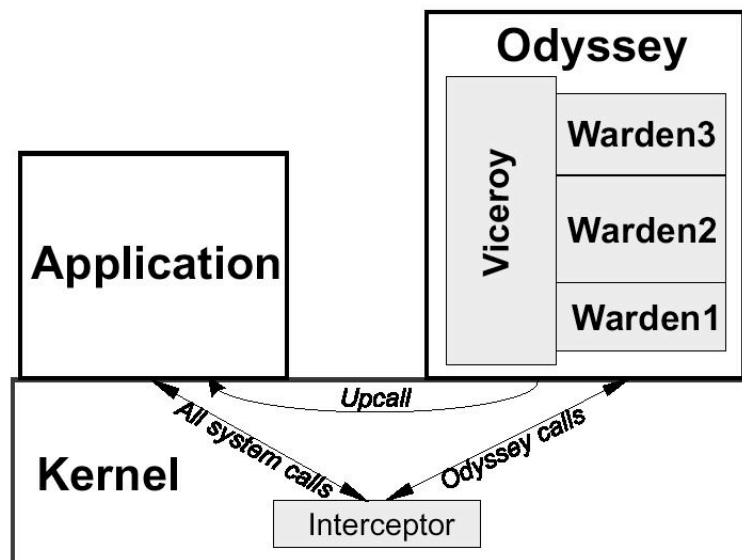
# Design & Implementation

- Integrated into NetBSD as a new VFS file system.
- New system calls included
- Odyssey (viceroy & wardens) implemented in user space outside the kernel
- odyssey system calls are redirected via an interceptor module in the kernel.



# Design & Implementation

---



- Integrated into NetBSD as a new VFS file system.
- New system calls included
- Odyssey implemented in user space outside the kernel, odyssey system calls are redirected via an interceptor module in the kernel.

# Design & Implementation

---

- Viceroy
  - A type independent component, responsible for centralized resource management
- Warden
  - A warden encapsulates system level support for a client
  - To support a new data type, an appropriate *warden* has to be written and incorporated into Odyssey at each client

# Design & Implementation

---

- Data-centric Adaptation
  - Between *viceroy* and *wardens*
  - It defines the fidelity levels for each data type and factors them into resource management
- Action-centric
  - Between applications and Odyssey
  - It provides applications with control over the selection of fidelity levels supported by the wardens

# API

---

- Applications communicate resource expectations (window) to Odyssey using the request system call
- If availability of resource is within window the viceroy registers it and returns a unique identifier (handler) for it, else an error is returned with the available resource level to the application.

request(in path, in resource-descriptor, out request-id)

cancel(in request-id)

# API

---

- an upcall to the application is generated by the viceroy when resource availability strays outside a registered window of tolerance.
  - Upon receiving the upcall, the application adjusts the fidelity according to its policies, and issues another request call with the revised window.
- tsop (type-specific operation) calls are used when an application needs to request fidelity changes.

handler(in request-id, in resource-id, in resource-level)

tsop(in path, in opcode, in insize, in inbuf, inout outside, out outbuf)

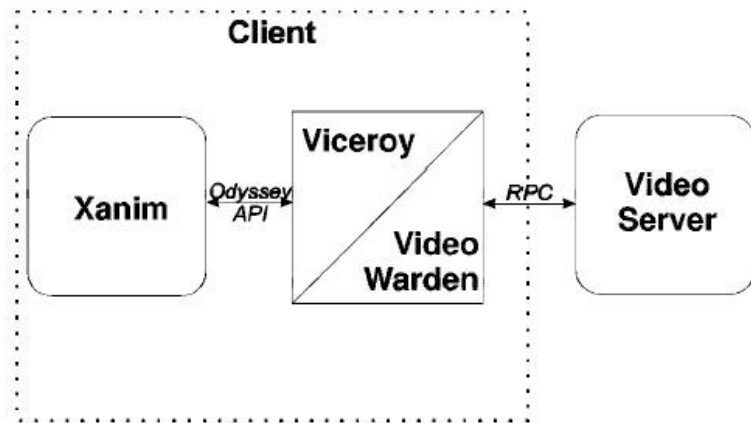
# Example Applications

---

- Video Player
- Web Browser
- Speech Recognizer

# Video Player (xanim)

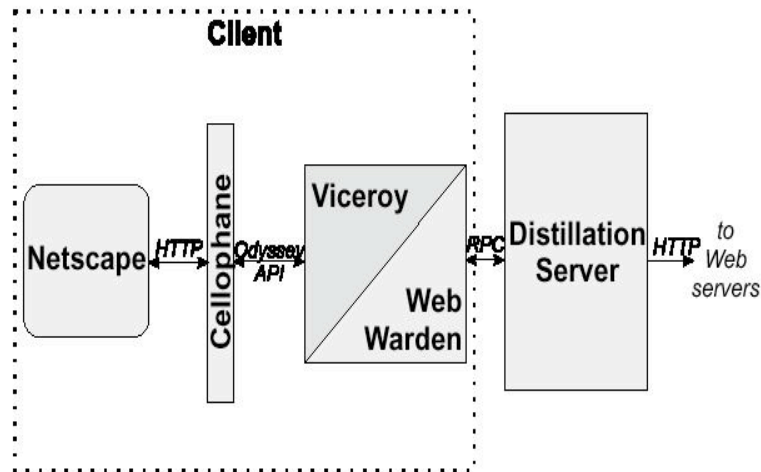
---



- The server stores a number of tracks of the movie, each with a different fidelity
- Number of tracks, the size and offset of frames for each track is stored in meta data

# Web Browser (Netscape)

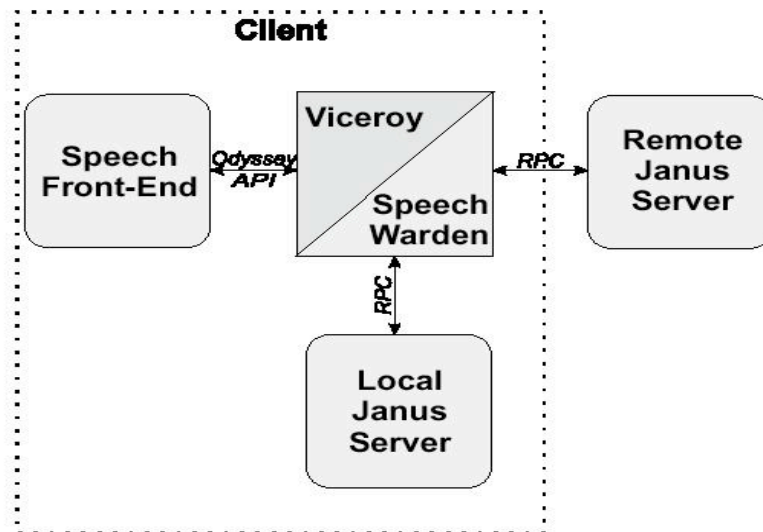
---



- The cellophane transforms the http requests into file operations on odyssey web objects
- The Web Warden is responsible for setting the fidelity level
- The distillation Server provides multiple levels of fidelity.

# Speech Recognizer

---



- Speech Warden is responsible for choosing to do a local, remote or hybrid (1<sup>st</sup> pass on client) recognition.
- Decision is dependent on available bandwidth

# Evaluation

---

- Criteria
  - How agile is Odyssey?
  - How beneficial is it for applications to exploit the dynamic adaptation?
  - How important is centralized resource management for concurrent resource management?
- Results
  - “Odyssey drops a factor of 2 to 5 fewer frames than the other strategies, and Web pages are loaded and displayed roughly twice as fast. The resulting decrease in network utilization improves speech recognition time as well.”