

IT Came From The Depths of Winter

# metaBot

Nicholas Roy  
Information Technology Services  
The University of Iowa

CIC Tech Forum 2009  
October 7<sup>th</sup>, 2009



SnowBot image copyright Think Geek

# The Identity Service at Iowa

- Powered by Microsoft Identity Lifecycle Manager (ILM)
- ILM's forte is to provision and synchronize directories
- Provisioning a *service* typically requires several steps, in a workflow
- Abstract and isolate this complexity from ILM to allow fast change processing

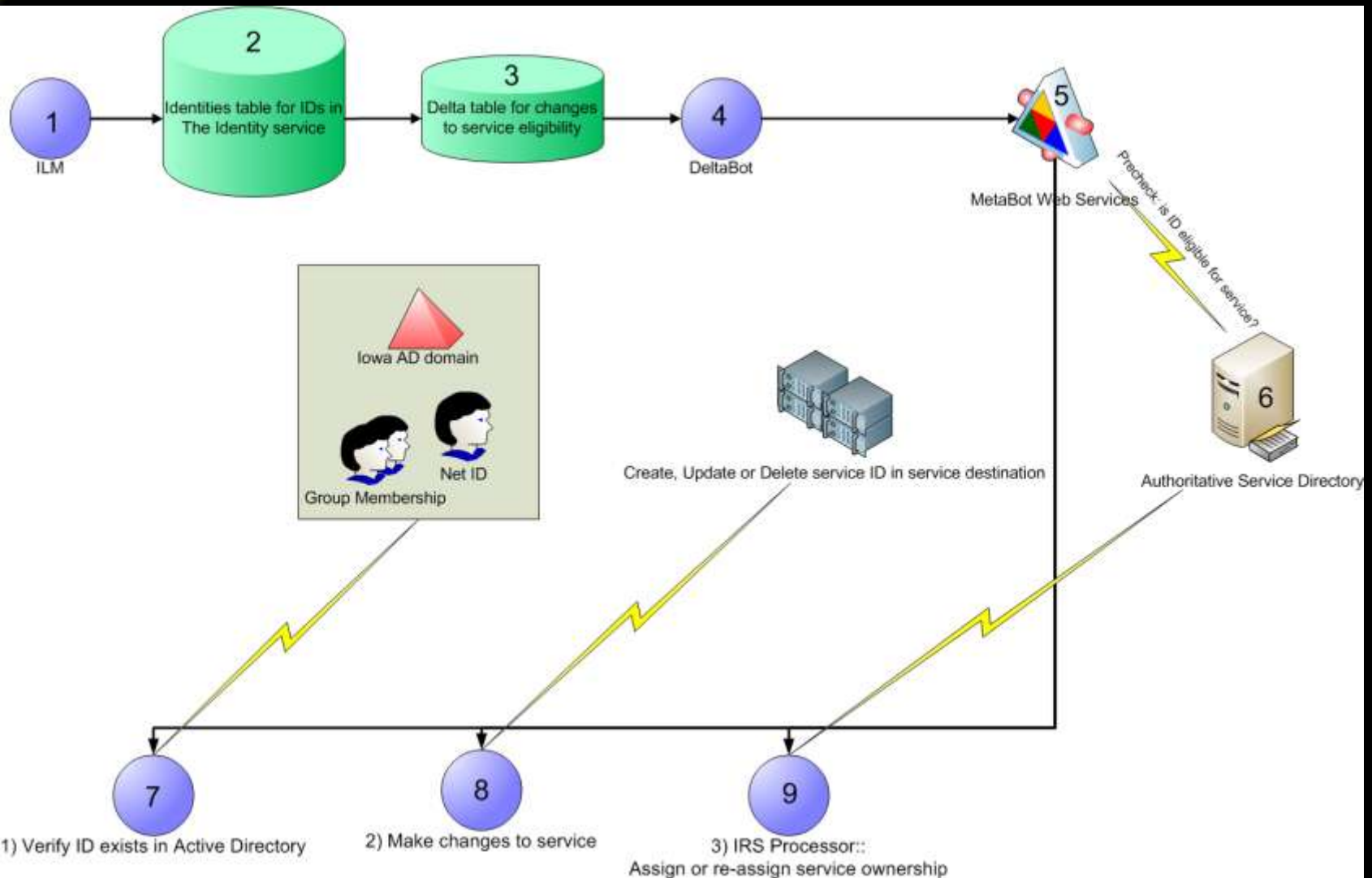
# Making Provisioning Work

- Goals
  - Loosely coupling service provisioning to the Identity service
  - Allow provisioning of both entitlement (based on role) and subscription (on-request) services
- Requirements
  - Accept requests 24x7
  - Reliable
  - Rules based
  - Validated
  - Sequenced

# Our Solution: MetaBot

- Robust service provisioning framework
- Load balanced, failover enabled
- REST (plain ol' XML) web service
- Stateful
- Rules based
- Validated
- Sequenced

# MetaBot



# Technologies Used

- Microsoft Identity Lifecycle Manager (ILM)
- Microsoft .NET Framework 3.5
- Windows Communication Foundation (WCF)
- ADO.NET Entity Framework (LINQ to Entities)
- Microsoft SQL Server 2005
- Microsoft Active Directory Application Mode (ADAM) LDAP directory

# Reliability

- Web services servers fronted by an F5 BigIP- we can receive service requests on a 24x7 basis and queue them
- SQL backing database is clustered for reliability
- ADAM directories are replicated for reliability

# Service Eligibility Changes

- Attribute changes in the identity service are reflected in a SQL table via an ILM management agent
- A trigger on this table generates service eligibility “delta” records in a delta table
- Two types of service eligibilities: entitlement and subscription
- “DeltaBot” process runs regularly to deliver these changes to MetaBot

# Service Deltas and DeltaBot

- DeltaBot calls the appropriate “web method” for MetaBot-handled services
- MetaBot’s REST Web methods (CRUD)
  - Create
  - Update (with multiple types of updates)
  - Delete

# Web Service – AD Security

- Authentication and Authorization
  - Based on Microsoft Active Directory group membership
  - Service ID calling the web service is vetted against AD groups named for the service and action being performed (e.g., create-filespace)

# Web Service Rules

- Input Validation
  - Parameters specified in a SQL table with maximum length
  - Rules for certain parameters are compiled into an extension library and marked to be run for each action and each relevant parameter
  - Mappings are stored in MetaBot's SQL database
  - Eligibility, parameter formatting, etc., checked

# Web Service Workflow

- Changes are broken up into steps
- Steps for each action for each service are determined in the MetaBot SQL database
- Parameters for each step are serialized into XML and committed to the database
- Each step has a unique identifier and an order in which it is to occur for each action for each service
- Change processors process these steps in order

# Change Processor Design

- Once written, can be used in multiple services and multiple service actions
- Pull change steps from the MetaBot database via a second “MetaBot Work” REST web service that ensures only one processor is working on a specific change at any one time, and that changes happen in the correct sequence

# Change Processor Design

- Programming language agnostic
- HTTP POST XML via SSL/port 443
- Processors can handle different priority requests
- Receive service name, action type and any relevant parameters for the change they are handling
- Call the MetaBot Work web service with their own unique identifier to check out their changes

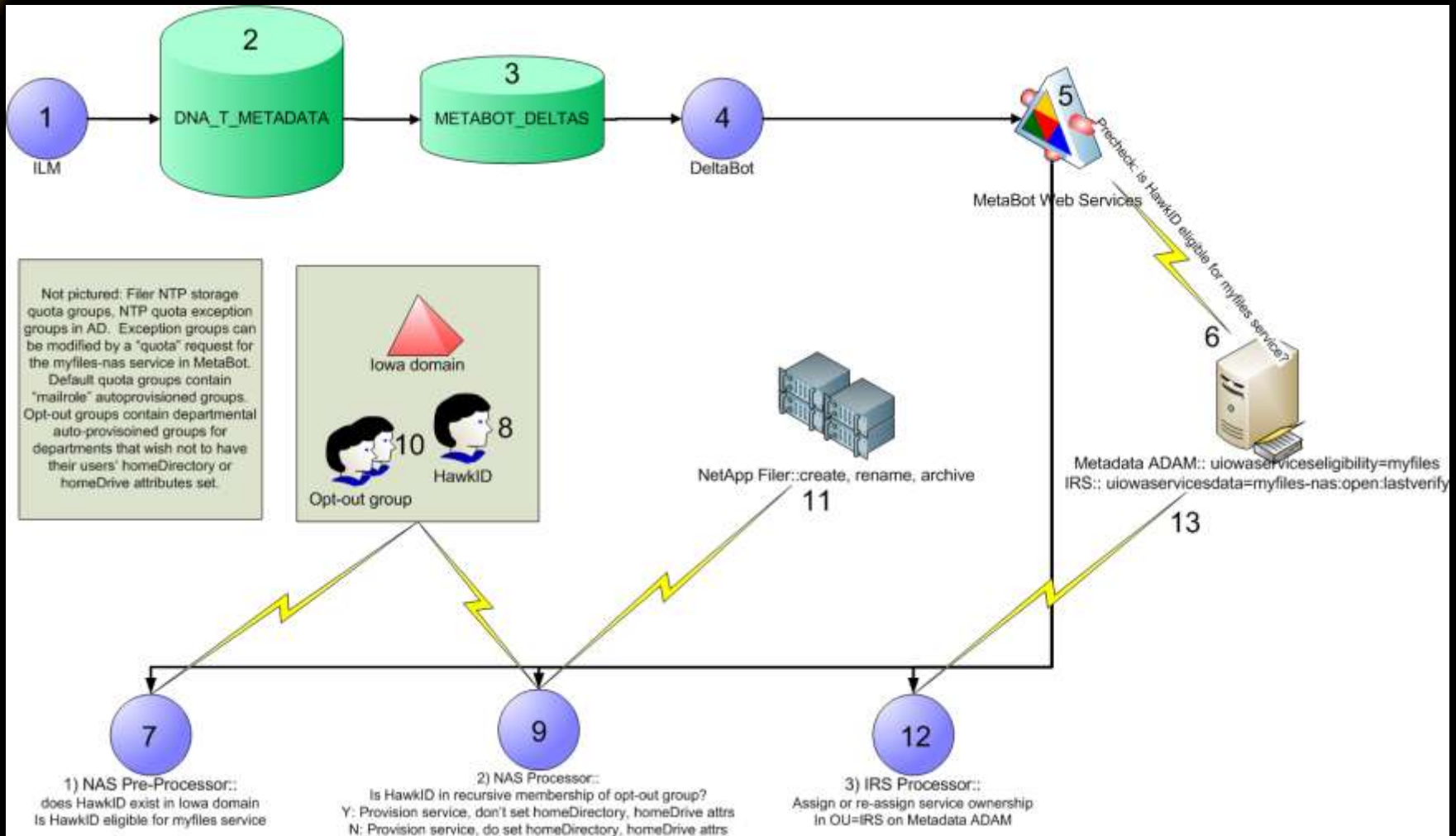
# Change Processor Reusability

- Example: IRS (ID Reservation System)
  - LDAP directory containing service records (service IDs linked to master record ID of the service ID owner)
  - Service records contain
    - Services for an ID
    - Service state (open date, last “verify” date)
    - Service expiration date
    - Service notification date (date of last notification of close of service)

# Change Processor Reusability

- IRS (continued)
  - IRS needs to be updated for every create, update or delete action for all tracked services
  - The IRS change processor can be added as a final step in MetaBot's change sequence for it to reconcile service records

# Example: myfiles-nas service



# Results

- Framework handles entitlement service changes generated via ILM
- Framework supports subscription requests from clients such as web applications
- Service processors process changes independently
- Service changes can be performed reliably and retried on failure

# Results

- Monitoring of the MetaBot SQL tables allows notification of any hung changes
- Have the ability to handle rollbacks due to tracking of previous state for all changes

# Summary

- Currently supporting one service (myfiles-nas)
- In production since the end of spring semester, 2009
- Goal is to support entitlement and subscription provisioning going forward using this framework