# Usage Control (UCON)

ISA 767, Secure Electronic Commerce
Xinwen Zhang, xzhang6@gmu.edu
George Mason University

## Access Control

- Origin
  - Since the advent of timesharing system
- The main goal is to selectively determine
  - who can access services, resources, and digital contents and
  - exactly what access is provided

# Access Control Models

- Evolvement of AC Models
  - Identity-based
    - AC Matrix, DAC, etc
  - Label-based
    - MAC
  - Function/duty/task/role-based:
    - RBAC, etc
  - Attribute-based:
    - UCON
    - DRM
    - Trusted Management



# Traditional Access Control

- To protect computer/information resources by limiting previously known users' actions or operations
- Access matrix based approach still remains unchanged (ACL, Capability list)
- Right is pre-defined and granted to a subject
- MAC, DAC, RBAC

# Trust Management

- TM deals with authorization process in distributed systems environment for the access of users who are previously unknown to the system
- Trust management does not utilize identity of a subject for authorization process. Rather, it utilizes capabilities or properties of a subject for authorization decisions.
- Only server-side information can be protected.



# Digital Rights Management (DRM)

- Superdistribution
- It's a system, a technology, a service, an application software, and a solution
- No concrete definition.
  - Many interests groups, many vendors, many solutions, but no standards
- Controlling and tracking access to and usage (including dissemination) of digital information objects
- Securing digital object itself, not the transmission
  - By using cryptographic, and watermarking technologies
- Business perspectives
  - Not just for protections, but new business models
  - Increased revenue

## DRM (continued)

- Problem-specific enhancement to traditional access control
- enables controls on usage of digital objects at client-side by utilizing Client-side reference monitor
- mainly focus on intellectual property rights protection.
- Architecture and Mechanism level studies, Functional specification languages – Lack of access control model

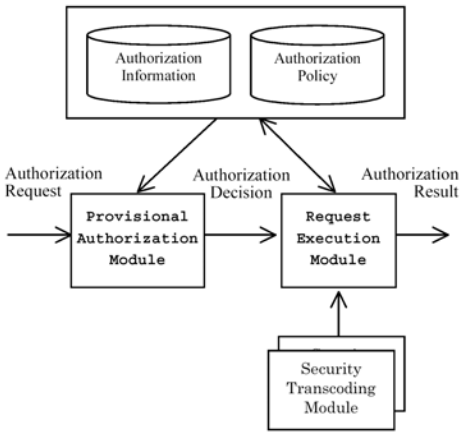


## And other works

- Incrementally enhanced models
  - Provisional authorization [Kudo & Hada, 2000]
  - EACL [Ryutov & Neuman, 2001]
  - Task-based Access Control [Thomas & Sandhu, 1997]
  - Ponder [Damianou et al., 2001]

# Provisional authorization

- Kudo & Hada, CCS'00
- An access can be authorized provided the subject (and/or the system) takes certain security actions :
  - You are allowed to access confidential information, but the access must be logged
  - You are allowed to read sensitive information, but you must sign a terms and conditions statement first
  - If unauthorized access is detected, a warning message must be sent to an administrator






# EACL [Ryutov & Neuman, 2001]

- Support of the advanced policies that allow actions when security violations are suspected or detected.
- Support policy enforcement at various time stages of the requested action.
- Simplify integration of related security services, such as authentication, intrusion detection, audit and notification with applications.
- Facilitate authorization decisions for applications.
- Provide generic policy evaluation environment.
- Provide a uniform integration model.
- Aim for extensibility to avoid the need to redesign the system in the future.

# EACL

- Tom can run a process on host bom.isi.edu.
  - If the request fails, a notification must be sent to a system administrator.
  - The process must not consume more than 20% of the CPU.
  - An audit record about the completed process must be generated.
- Conditions
  - Identity, authentication method,
  - Payment, Time,
  - Location
  - Notification
  - Audit System Threat Level Threshold
  - Application specific
- Continuous control and update issues



# Task-based

- Task-based Access Control [Thomas & Sandhu, 1997]
- Consumable rights
- Authorization is one-time and request-based permission by utilizing consumable rights.

# Ponder

- A policy language
  - Authorizations
  - Obligations (more like duties)
  - Delegations

```
type auth+  printing (subject S, target T, int validfrom, int validto, int maxPages) {
            action T.print (document);
            when time.between (validfrom, validto) && document.size () <= maxPages;
        }

        inst auth+  printingpolicy = printing ( /secretaries, /printers/colour, 0900, 1700, 10);

    type oblig+  printManagement (subject S, target T) {
                 on printError (printer, error);
                 do T.notify (printer, error) ->  S.log (printer, error);
    }

    inst oblig  p2 = printManagement (/printManager, /operators);
```



# Problem Statement (1)

- Traditional access control models are not adequate for today's distributed, network-connected digital environment.
  - Authorization only – No obligation or condition based control
  - Decision is made before access – No ongoing control
  - No consumable rights - No mutable attributes
  - Rights are pre-defined and granted to subjects

# Problem Statement (2)

- No access control models available for DRM.
- Recently enhanced models are not comprehensive enough to resolve various shortcomings.
- Need for a unified model that can encompass traditional access control models, DRM and other enhanced access control models from recent literature



# Motivations

- Highly dynamic and distributed computing environments require flexible AC
- Object can be located in various places
  - General client side platforms
- Unknown or partial authenticated users
  - General attributes of users

# Motivations

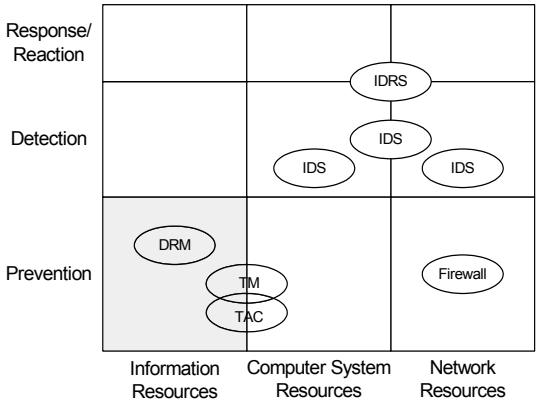
- Multi-aspects of access control decisions
  - Attributes of subjects and objects
  - Obligations
  - Environmental conditions
- Continually control
  - Access is has a duration - usage
- Dynamics of subject and object attributes



# Security Techniques

- **Prevention**
  - **access control**
- **Detection**
  - **auditing/intrusion detection**
  - **incident handling**
  - **Tracing**
- **Response/Reaction/Recover**
  - **Backup**
  - **Restore**
- **Acceptance**
  - Tolerance and practicality

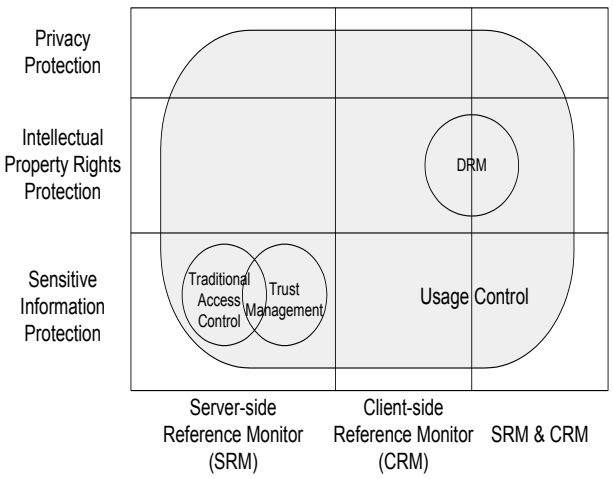
# Research Scope in Infosec




- Security Objectives
  - Prevention
  - Detection
  - Response/reaction
- Target Resources
  - Information resources
  - Computer system resources
  - network resources

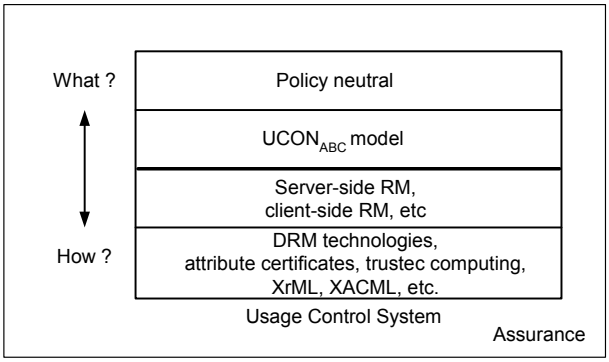


# Usage Control (UCON) Coverage



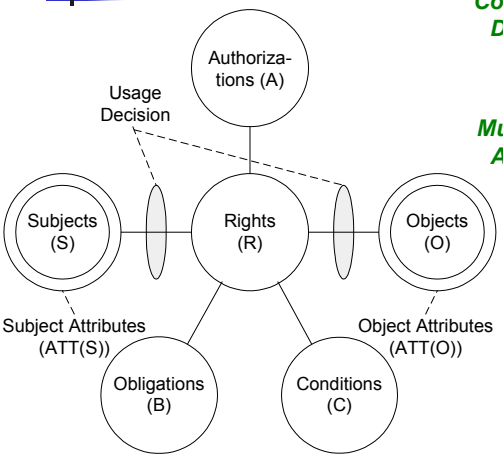

- Protection Objectives
  - Sensitive information protection
  - IPR protection
  - Privacy protection
- Protection Architectures
  - Server-side reference monitor
  - Client-side reference monitor
  - SRM & CRM

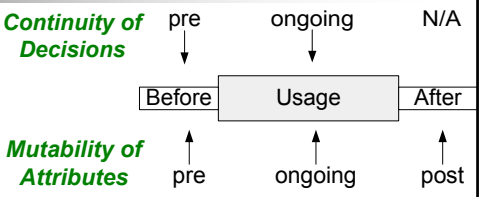
# OM-AM layered Approach






# Building UCON$_{ABC}$ Models




- Continuity
  - Decision can be made during usage for continuous enforcement
- Mutability
  - Attributes can be updated as side-effects of subjects' actions

# Subjects (S)

- entities associated with attributes, and hold and exercise certain rights on objects
- For simplicity, subject can be regarded as representing an individual human being
- Consumer, Provider, Identifiee subjects
  - Identifiee subjects: identified subjects in digital objects that include their privacy-sensitive information. (patients in health care system).



# Subject Attributes (ATT(S))

- Properties of a subject that can be used for the usage decision process
- identity, role, credit, membership, security level, capability, etc.
- Immutable attributes: can be changed only by administrative action
- Mutable attributes: can be modified as a side effects of subject's access to objects (credit, clearance with high watermark, access time, etc.)
- Trusted source of attribute values and timeliness is prerequisite for UCON.

# Objects (O)

- Entities that subjects hold rights on.
- associated with attributes, either by themselves or together with rights.
- Security sensitive objects
- Privacy sensitive objects
- IP objects
- Original vs. derivative objects
  - A derivative object is created in consequence of obtaining or exercising rights on an original object. (usage log, payment information, etc.)



# Object Attributes (ATT(O))

- Properties of an object that can be used for the usage decision process
- Security classification, role, price, etc.
- Immutable and mutable attributes

# Rights (R)

- A subject's privilege on an object
- A set of usage functions that enables a subject's access to objects
- May or may not have a hierarchy
- Existence of right is determined when access is attempted by a subject (not by a predefined access matrix)
- Delegation of rights and administrative rights are not covered here.
  - Distinguished from rights from subject to objects.



# Three Decision Factors Two Decision Properties

- 3 Decision Factors
  - Authorizations (A)
  - oBligations (B)
  - Conditions (C)
  - A,B, and C are functional predicates used for usage decision making.
- 2 Decision Properties
  - Mutability
  - Continuity

# Authorizations (A)

- Functional predicates that have to be evaluated for usage decision based on subject and object attributes and the requested specific right
  - *preA*: decision is made prior to the access
  - *onA*: decision is made during the access
    (e.g., Certificate Revocation List (CRL))
- Updates on Attributes: pre, ongoing, post
  - *preUpdate*: High watermark policy
  - *onUpdate*: Pre-paid credit for time-based metering
  - *postUpdate*: Metered usage payment



# oBligations (B)

- Functional predicates that verify mandatory requirements a subject has to perform before or during a usage exercise.
  - *preB* utilizes history function to check if certain activities have been fulfilled or not. (a user have to fill out personal info to download a white paper)
  - *onB* predicate has to be satisfied continuously during usage. (a user has to watch an ad window while using free Internet services)
    - Continuously
    - Periodically
    - conditionally
- Updates on Attributes: *preUpdate, onUpdate, postUpdate*

# Conditions (C)

- Evaluate current environmental or system status for usage decision
  - *preC*: condition is checked before usage
  - *onC*: condition has to be satisfied while usage
- Attributes can be used to select which condition requirements has to be satisfied
- No attribute updates
- Time period (Office hour), location (area code, CPU-id, IP address), system status (normal, high alert, under attack), etc.



# Mutability and Continuity

- With continuity property, decision can be made even after access is allowed.
- Mutability means mutability of attributes. So, With mutability property, attributes can be either immutable or mutable.
  - Immutable attributes can be modified only by administrative actions
  - Mutable attributes can be modified as side-effects of subjects' actions

# Examples

- Long-distance phone (pre-authorization with post-update)
- Pre-paid phone card (ongoing-authorization with ongoing-update)
- Pay-per-view (pre-authorization with pre-updates)
- Click Ad within every 30 minutes (ongoing-obligation with ongoing-updates)
- Business Hour (pre-/ongoing-condition)

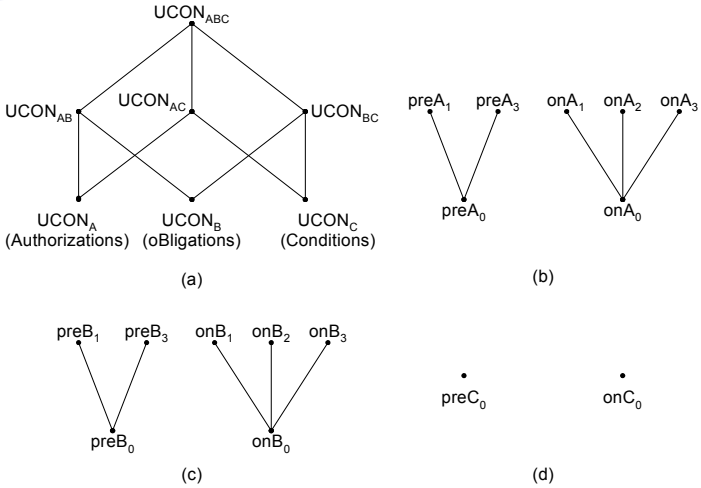


# $UCON_{ABC}$ Model Space

| | 0(Immutable) | 1(pre) | 2(ongoing) | 3(post) |
|---|---|---|---|---|
| preA | Y | Y | N | Y |
| onA | Y | Y | Y | Y |
| preB | Y | Y | N | Y |
| onB | Y | Y | Y | Y |
| preC | Y | N | N | N |
| onC | Y | N | N | N |

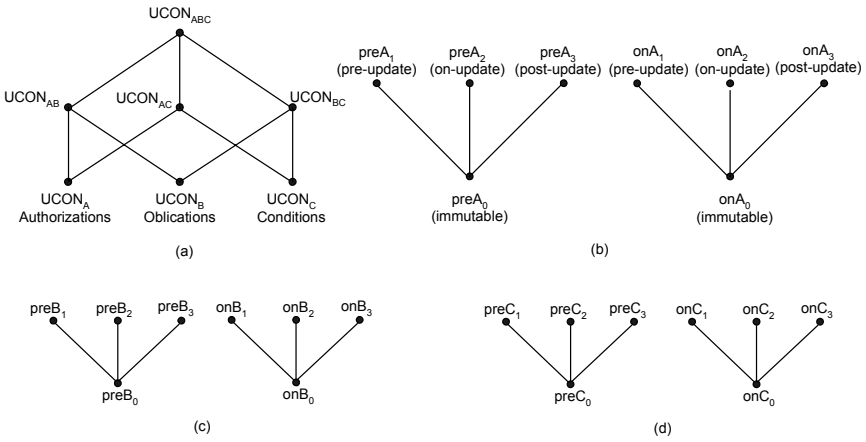
N : Not applicable

# A Family of $UCON_{ABC}$ Core Models

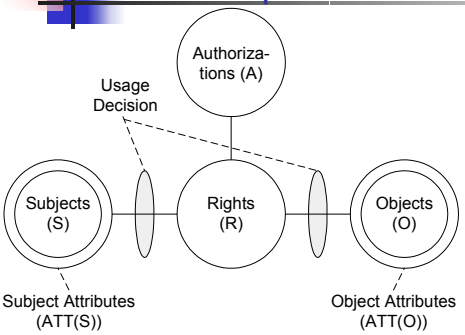


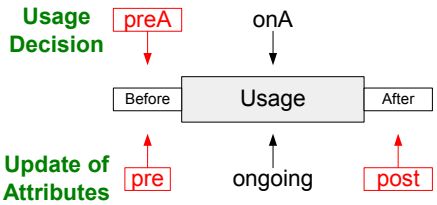



# Family of Core Models (recent)



# UCON$_{preA}$

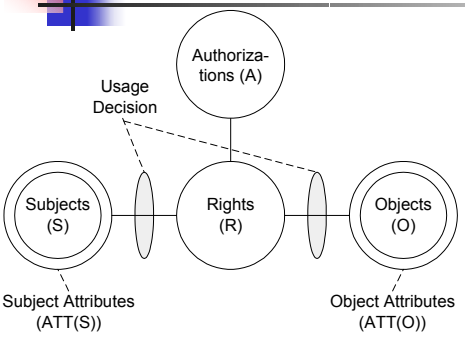


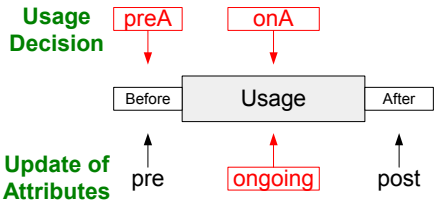




- Online content distribution service
  - Pay-per-view (pre-update)
  - Metered payment (post-update)



# UCON$_{onA}$




- Pay-per-minutes (pre-paid Phone Card)

# $UCON_{preA}$: pre-Authorizations Model

- $UCON_{preA0}$
  - *S, O, R, ATT(S), ATT(O)* and *preA* (subjects, objects, rights, subject attributes, object attributes, and pre-authorizations respectively);
  - *allowed(s,o,r)* $\Rightarrow$ *preA(ATT(s),ATT(o),r)*
- $UCON_{preA1}$
  - *preUpdate(ATT(s)),preUpdate(ATT(o))*
- $UCON_{preA3}$
  - *postUpdate(ATT(s)),postUpdate(ATT(o))*



# $UCON_{preA0}$: MAC Example

- *L* is a lattice of security labels with dominance relation $\geq$
- *clearance:* $S \rightarrow L$
- *classification:* $O \rightarrow L$
- *ATT(S) = {clearance}*
- *ATT(O) = {classification}*
- *allowed(s,o,read)* $\Rightarrow$ *clearance(s)* $\geq$ *classification(o)*
- *allowed(s,o,write)* $\Rightarrow$ *clearance(s)* $\leq$ *classification(o)*

## DAC in UCON: *with ACL ($UCON_{preA0}$)*

- *N* is a set of identity names
- *id : S* $\rightarrow$ *N*, one to one mapping
- *ACL : O* $\rightarrow 2^{N \times R}$, *n* is authorized to do *r* to *o*
- *ATT(S)= {id}*
- *ATT(O)= {ACL}*
- *allowed(s,o,r)* $\Rightarrow$ *(id(s),r)* $\in$ *ACL(o)*



## RBAC in UCON: *$RBAC_1$ ($UCON_{preA0}$)*

- *P = {(o,r)}*
- *ROLE* is a partially ordered set of roles with dominance relation $\geq$
- *actRole: S* $\rightarrow 2^{ROLE}$
- *Prole: P* $\rightarrow 2^{ROLE}$
- *ATT(S) = {actRole}*
- *ATT(O) = {Prole}*
- *allowed(s,o,r)* $\Rightarrow \exists$*role* $\in$ *actRole(s)*, $\exists$*role'* $\in$ *Prole(o,r)*, *role* $\geq$ *role'*

## DRM in UCON: *Pay-per-use with a pre-paid credit ($UCON_{preA1}$)*

- *M* is a set of money amount
- *credit: S → M*
- *value: O x R → M*
- *ATT(s): {credit}*
- *ATT(o,r): {value}*
- *allowed(s,o,r) ⇒ credit(s) ≥ value(o,r)*
- *preUpdate(credit(s)): credit(s) = credit(s) - value(o,r)*



## $UCON_{preA3}$ : DRM Example

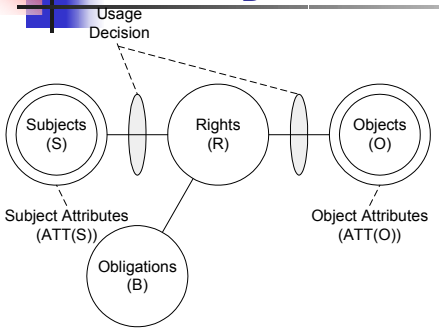
- Membership-based metered payment
  - *M* is a set of money amount
  - *ID* is a set of membership identification numbers
  - *TIME* is a current usage minute
  - *member: S → ID*
  - *expense: S → M*
  - *usageT: S → TIME*
  - *value: O x R → M* (a cost per minute of r on o)
  - *ATT(s): {member, expense, usageT}*
  - *ATT(o,r): {valuePerMinute}*
  - *allowed(s,o,r) ⇒ member(s) ≠ ∅*
  - *postUpdate(expense(s)): expense(s) = expense(s) + (value(o,r) x usageT(s))*

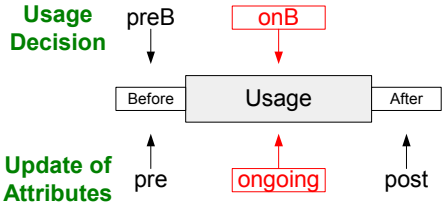
# $UCON_{onA}$: ongoing-Authorizations Model

- $UCON_{onA0}$
  - *S, O, R, ATT(S), ATT(O)* and *onA*;
  - *allowed(s,o,r)* $\Rightarrow$ *true;*
  - *Stopped(s,o,r)* $\Leftarrow \neg$*onA(ATT(s),ATT(o),r)*
- $UCON_{onA1}$, $UCON_{onA2}$, $UCON_{onA3}$
  - *preUpdate(ATT(s)),preUpdate(ATT(o))*
  - *onUpdate(ATT(s)),onUpdate(ATT(o))*
  - *postUpdate(ATT(s)),postUpdate(ATT(o))*
- Examples
  - Certificate Revocation Lists
  - revocation based on starting time, longest idle time, and total usage time



# $UCON_B$




- Free Internet Service Provider
  - Watch Ad window (no update)
  - Click ad within every 30 minutes (ongoing update)

# $UCON_{preB0}$: pre-oBligations w/ no update

- *S, O, R, ATT(S),* and *ATT(O)*;
- *OBS, OBO* and *OB* (obligation subjects, obligation objects, and obligation actions, respectively);
- *preB* and *preOBL* (pre-obligations predicates and pre-obligation elements, respectively);
- $preOBL \subseteq OBS \times OBO \times OB$;
- $preFulfilled: OBS \times OBO \times OB \rightarrow \{true,false\}$;
- $getPreOBL: S \times O \times R \rightarrow 2^{preOBL}$, a function to select pre-obligations for a requested usage;
- $preB(s,o,r) = \Lambda_{(obs\_i,obo\_i,ob\_i) \in getPreOBL(s,o,r)}\ preFulfilled(obs_i,obo_i,ob_i);$
- $preB(s,o,r) = true$ by definition if $getPreOBL(s,o,r)=\varnothing$;
- $allowed(s,o,r) \Longrightarrow preB(s,o,r)$.

- Example: License agreement for a whitepaper download

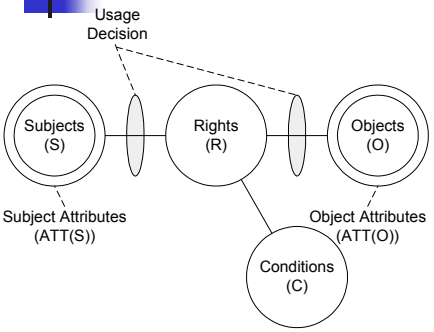


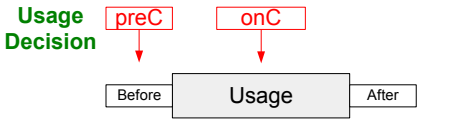
# $UCON_{onB0}$: ongoing-oBligations w/ no update

- *S, O, R, ATT(S), ATT(O), OBS, OBO* and *OB*;
- *T*, a set of time or event elements;
- *onB* and on*OBL* (on-obligations predicates and ongoing-obligation elements, respectively);
- $onOBL \subseteq OBS \times OBO \times OB \times T$;
- $onFulfilled: OBS \times OBO \times OB \times T \rightarrow \{true,false\}$;
- $getOnOBL: S \times O \times R \rightarrow 2^{onOBL}$, a function to select ongoing-obligations for a requested usage;
- $onB(s,o,r) = \Lambda_{(obs\_i,obo\_i,ob\_i,\ t\_i) \in getOnOBL(s,o,r)}\ onFulfilled(obs_i,obo_i,ob_i,t_i);$
- $onB(s,o,r) = true$ by definition if $getOnOBL(s,o,r)=\varnothing$;
- $allowed(s,o,r) \Rightarrow true;$
- $Stopped(s,o,r) \Leftarrow \neg\ onB(s,o,r).$

- Example: Free ISP with mandatory ad window

# $UCON_C$




**Update of Attributes:** **No-Update is possible**

- Location check at the time of access request
- Accessible only during business hours



# $UCON_{preC0}$: pre-Condition model

- *S, O, R, ATT(S),* and *ATT(O)*;
- *preCON* (a set of pre-condition elements);
- *preConChecked: preCON* $\rightarrow$ *{true,false}*;
- *getPreCON: S x O x R* $\rightarrow$ $2^{preCON}$;
- *preC(s,o,r)* = $\Lambda_{preCon\_i \in getPreCON(s,o,r)}$ *preConChecked(preCon$_i$);*
- *allowed(s,o,r)* $\Longrightarrow$ *preC(s,o,r)*.
- Example: location checks at the time of access requests

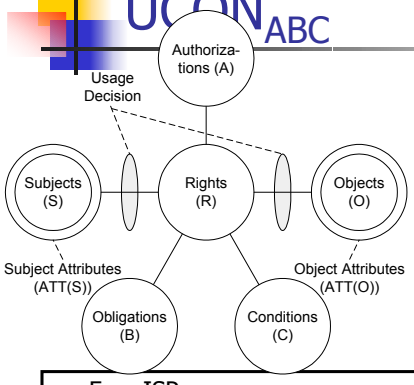
# $UCON_{onC0}$: ongoing-Condition model

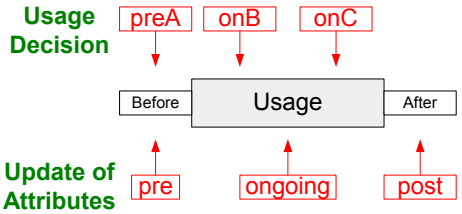
- *S, O, R, ATT(S),* and *ATT(O)*;
- *onCON* (a set of on-condition elements);
- *onConChecked: onCON* $\rightarrow$ *{true,false}*;
- *getOnCON: S x O x R* $\rightarrow 2^{onCON}$;
- *onC(s,o,r)* = $\Lambda_{onCon\_i \in getOnCON(s,o,r)}$ *onConChecked(onCon$_i$);*
- *allowed(s,o,r)* $\Rightarrow$ *true;*
- *Stopped(s,o,r)* $\Leftarrow \neg$*onC(s,o,r)*

- Example: accessible during office hour



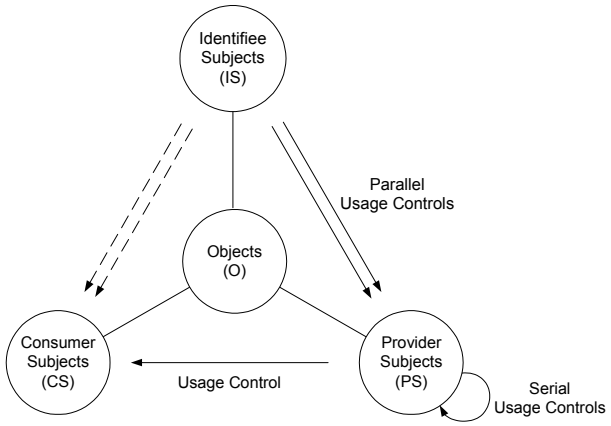
# $UCON_{ABC}$




- Free ISP
  - Membership is required (pre-authorization)
  - Have to click Ad periodically while connected (on-obligation, on-update)
  - Free member: no evening connection (on-condition), no more than 50 connections (pre-update) or 100 hours usage per month (post-updates)

# Beyond the $UCON_{ABC}$ Core Models






# Conclusion

- Developed A family of $UCON_{ABC}$ core models for Usage Control (UCON) to unify *traditional access control models, DRM*, and other modern enhanced models.
- $UCON_{ABC}$ model integrates *authorizations, obligations, conditions*, as well as *continuity* and *mutability* properties.

# Future Research

- Enhance the model
  - UCON administration or management
  - Detail of update procedure in $UCON_{ABC}$ model
  - Delegation of usage rights
- Develop Architectures and Mechanisms
  - Payment-based architectures
  - CRM and SRM
  - Architectures for multi-organizations (B2B)
- UCON Engineering
  - Analysis of policy
  - Designing/modeling rules and Attributes