WORLD WIDE WEB 2012 LYON - FRANCE

Multi-application Profile Updates Propagation: a Semantic Layer to improve Mapping between Applications

Nadia Bennani – *Université de Lyon*
Max Chevalier - *Université Paul Sabatier*
Elöd Egyed-Zsigmond – *Université de Lyon*
Gilles Hubert – *Université Paul Sabatier*
Marco Viviani – *Università degli Studi dell'Insubria*

MultiA-Pro 2012

APRIL 16TH 2012

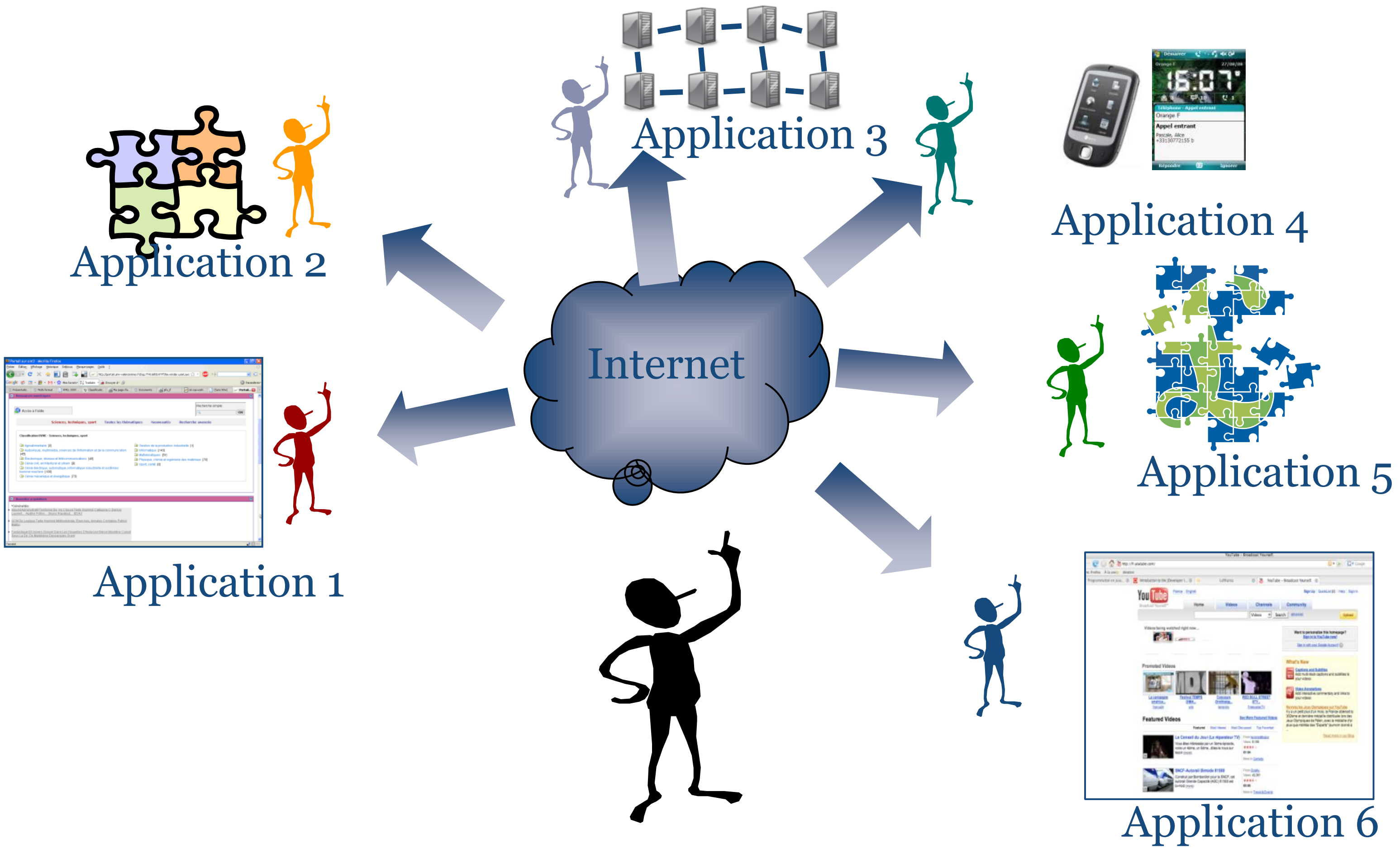
OUTLINE

- Open Issues in Multi-application Personalization
- G-Profile
- The Semantic Layer
- Benefits of Integration
- Conclusions and Further Research

Introduction

- Nowadays, many applications in different areas (digital libraries, search engines, e-learning, online databases, ecommerce, social networks…) collect information about users for service personalization.
- Applications organize user properties, preferences and assumptions based on the user state, in *user profiles*.
- Each application manages user information independently from others, using a specific *user model*.

Mono application user profile management



Application 2

Application 3

Application 4

Application 5

Internet

Application 1

Application 6

Drawbacks

- *Data incoherence* among isolated user profiles can be produced, due to several drawbacks strictly connected to mono-application personalization.
  - *Redundancy*.
  - *Lack of inter-application experience*: data connected to a given user remain private to each application. Users cannot take advantage of their information scattered across different applications.
  - *Lack of inter-user experience*: users cannot profit of the experience already accumulated by other users, in the same or different applications.
  - *Lack of control*: users have little or no control over the information defining their profiles.

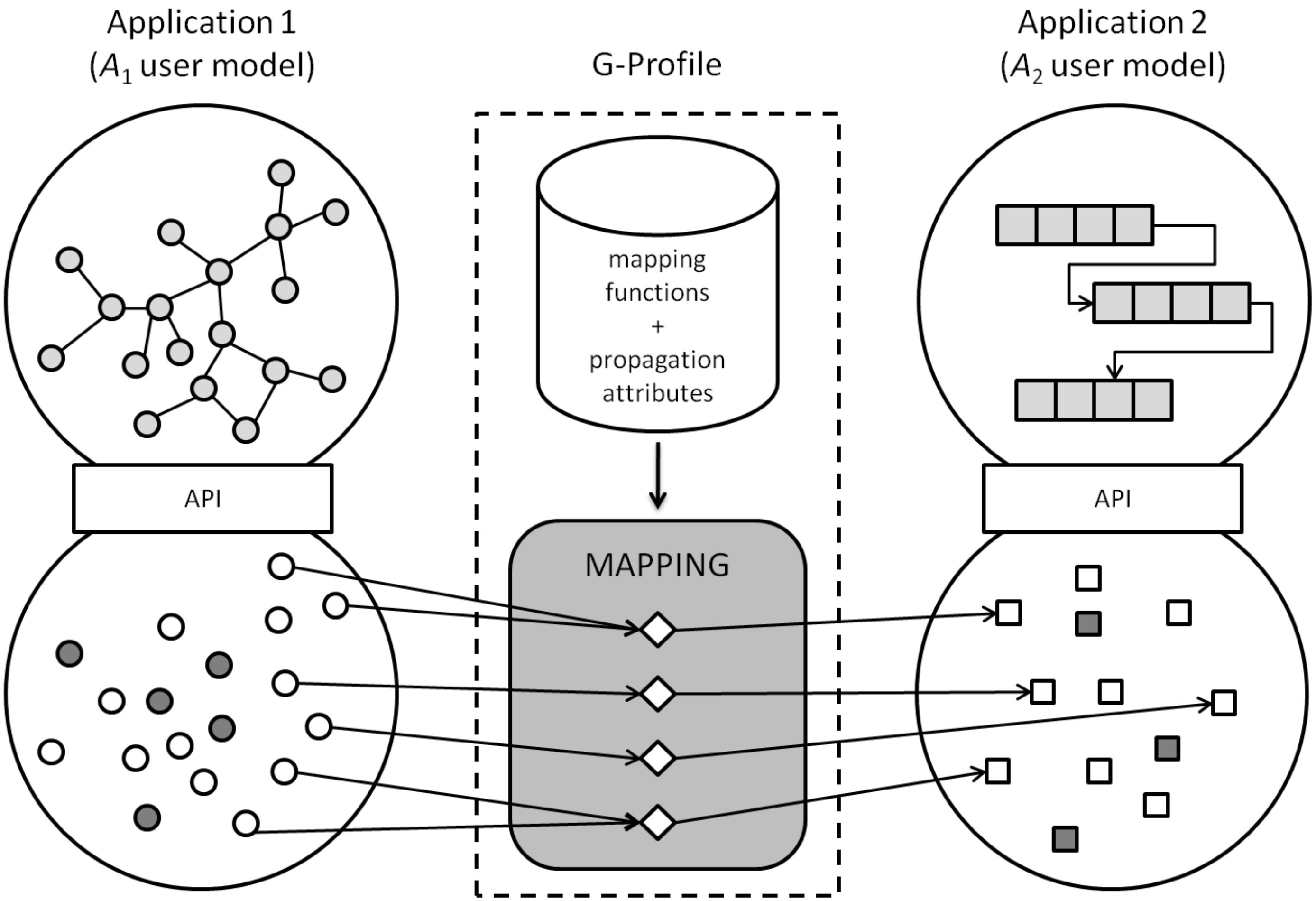
Aim of our Work

- *G-Profile*: a *multi-application user modeling system*
- G-Profile allows user profile information to *evolve* in a multi-application context by user data *propagation*.
- G-Profile is based on user profile mappings between applications.
- To improve mapping management and to limit human intervention, we propose to add to G-Profile a *Semantic Layer*: a module allowing to automatically identify these mappings.

G-Profile

- G-Profile does not propose neither a specific reconciliation technique able to take into account all the possible user data representations in different applications, nor a standard user profile model.
- We define some abstract *mapping functions*, based on the generic concept of *mapping between user data* among applications.
- An application is *G-Profile-aware* if it provides a suitable application programming interface (API) to access both its user profile attributes and a set of mapping functions for these attributes to be used in mapping generation assisted by G-Profile.

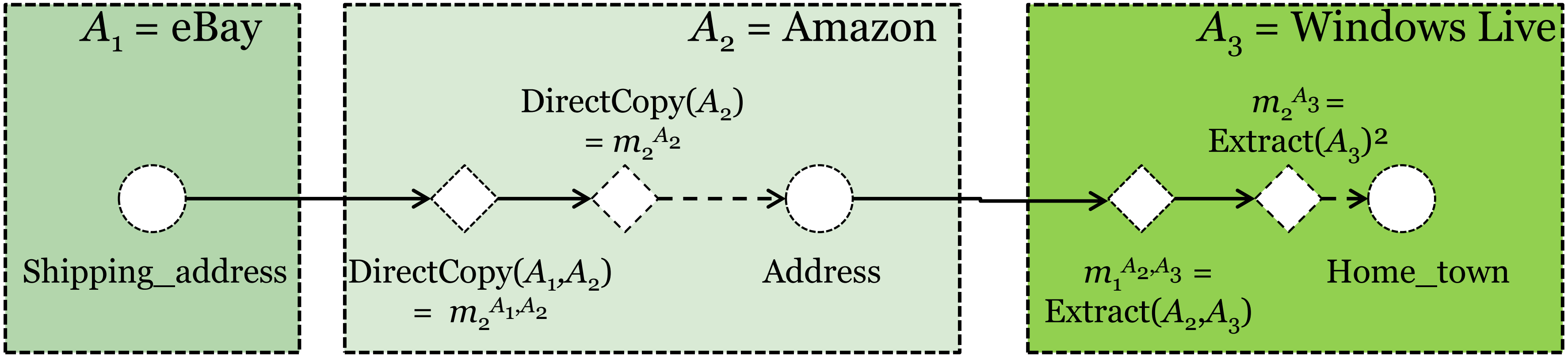
Architecture



User Profile Formalization

- Each application A manages a set of user attributes $a^A{}_k$
- $k \in \{1, ..., m_A\}$
- m_A is the total number of attributes for the application A
- for each user u_x using the application A, each attribute $a^A{}_k$ has a value v_k associated, forming the *user profile element* as a couple (*attribute, value*)
- Formally $e_k^{A,u_x} = \langle a_k^A, v_k \rangle$

Mapping example 1/2


A_1 = eBay
A_2 = Amazon
A_3 = Windows Live
DirectCopy(A_2) = $m_2^{A_2}$
$m_2^{A_3}$ = Extract(A_3)[2]
Shipping_address
DirectCopy(A_1,A_2) = $m_2^{A_1,A_2}$
Address
$m_1^{A_2,A_3}$ = Extract(A_2,A_3)
Home_town

Data Mapping Formalization – 1/2

- Each attribute can, from time to time, be involved as the *source* or the *target attribute* in a relation with others.

- More specifically, since attributes are organized differently in each application A_i depending on the adopted user model, they can be permuted in several *source sets*

$$S_l^{A_i} = \left\{ s_1^{A_i}, s_2^{A_i}, \ldots, s_{t_{A_i}}^{A_i} \right\}$$

Data Mapping Formalization – 2/2

- In the same way, each attribute of the application A_i can be a *target attribute* belonging to the *target set*

$$T^{A_i} = \left\{t_1^{A_i}, t_2^{A_i}, \dots, t_{v_{A_i}}^{A_i}\right\}$$

- We define a *mapping between two applications* A_i and A_j , $i \neq j$, as the triple

$$\mathcal{M}^{A_i,A_j} = \langle \mathbf{S}^{A_i}, T^{A_j}, M^{A_i,A_j} \rangle$$

- Formally a *mapping function* $m_k^{A_i,A_j} : S_l^{A_i} \to t_h^{A_j}$

Mapping Graph Formalization

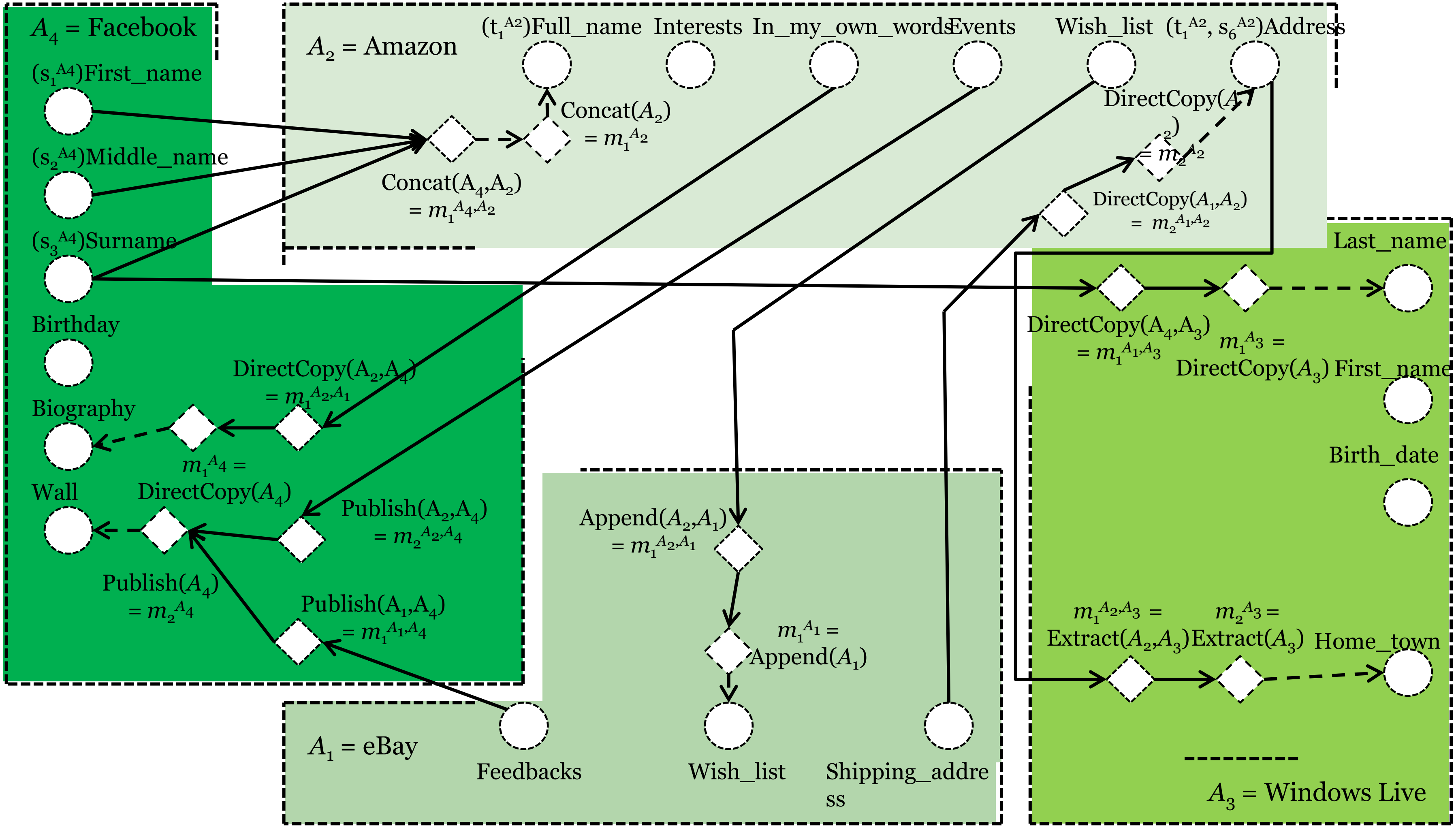
- It is possible to define a *mapping graph G* as a *combination of all the* mappings in our environment.

- *G* is a *directed graph G = (V,E)* composed of (*i*) a set *V* of *nodes,* (*ii*) a set *E* of *directed edges*.

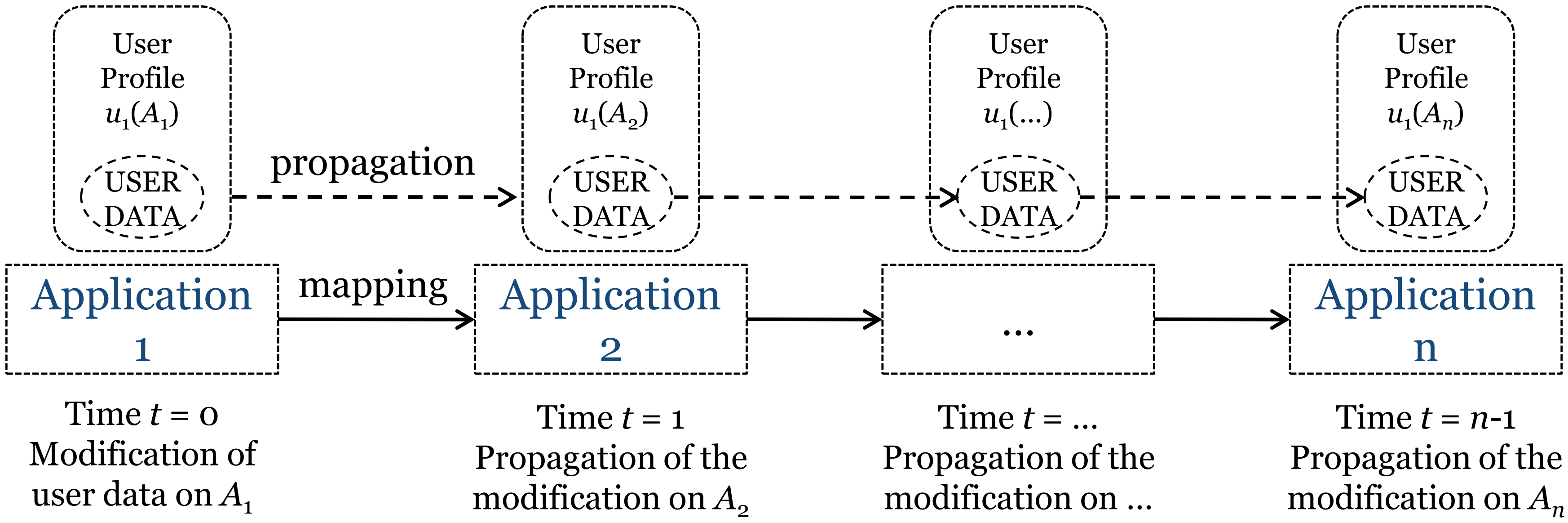
- We define two kinds of node: *attribute nodes* (*n-att*) and *function nodes* (*n-fun*). $V = V_{n\text{-}att} \cup V_{n\text{-}fun}$

- Formally $S_l^{A_i} \in V_{\texttt{n-att}}, \quad t_h^{A_j} \in V_{\texttt{n-att}}, \quad m_k^{A_i,A_j} \in V_{\texttt{n-fun}}$

Mapping example 2/2



Profile change propagation (Eg. 1)



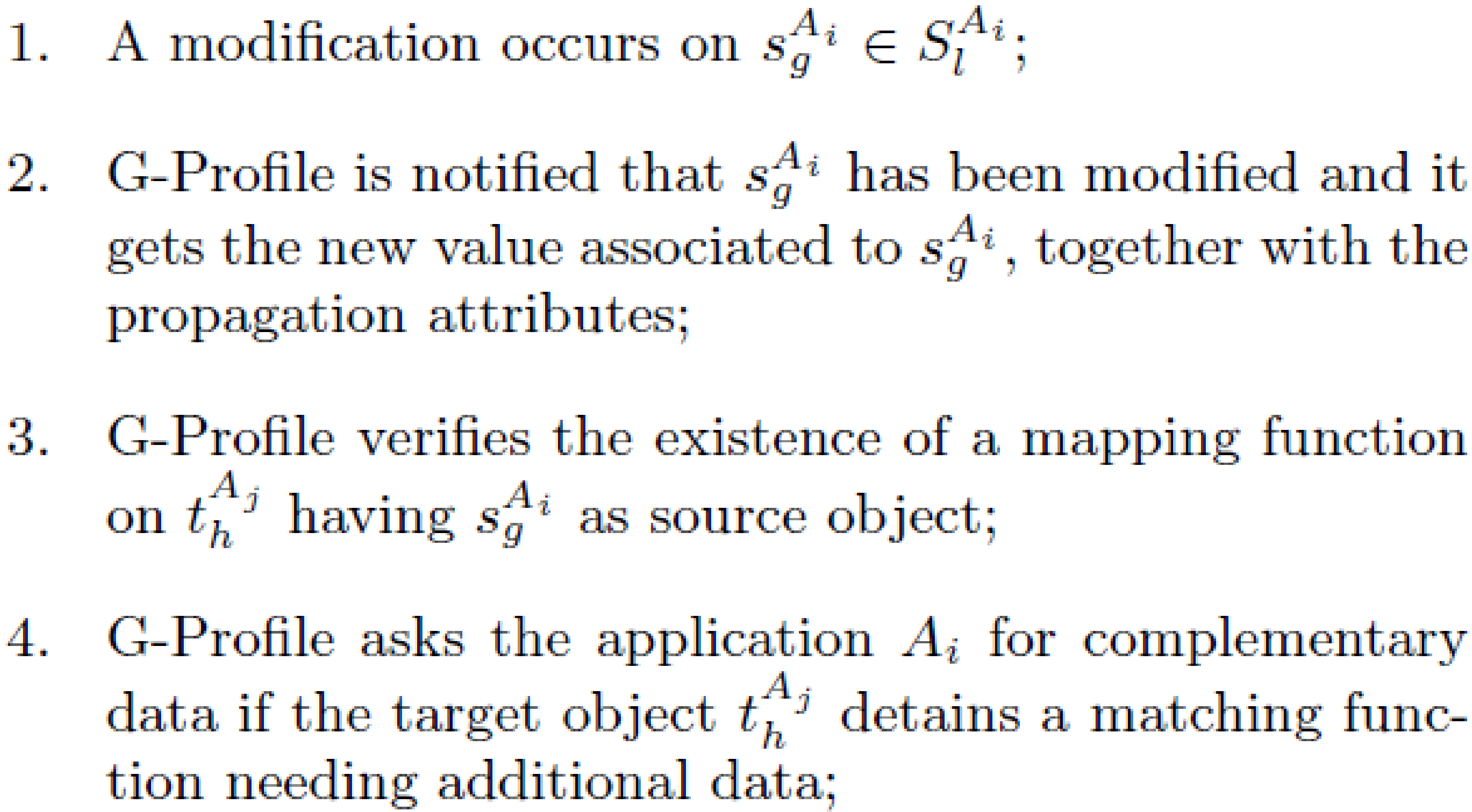
Time t = 0
Modification of user data on A_1

Time t = 1
Propagation of the modification on A_2

Time t = …
Propagation of the modification on …

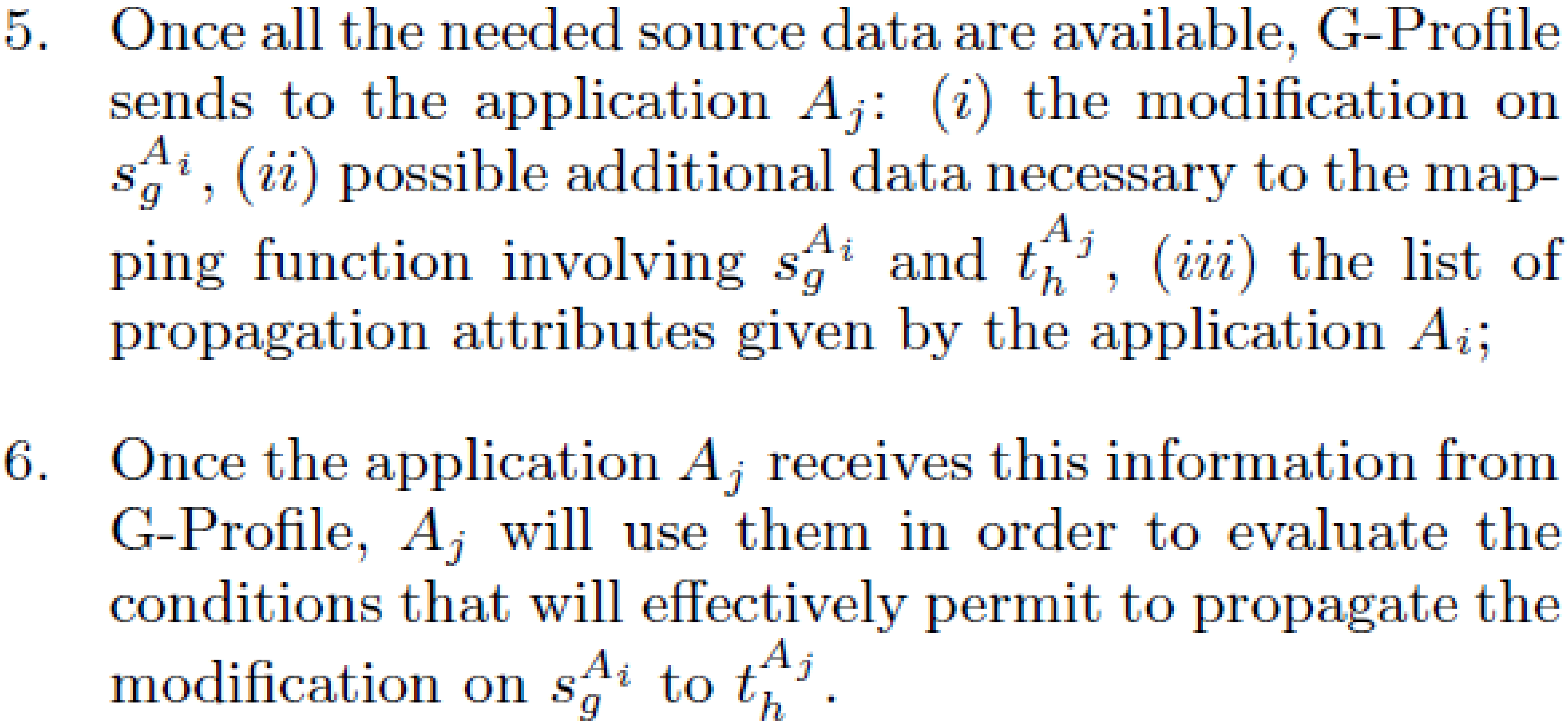
Time t = n-1
Propagation of the modification on A_n

Data Propagation – 1/2



1. A modification occurs on $s_g^{A_i} \in S_l^{A_i}$;
2. G-Profile is notified that $s_g^{A_i}$ has been modified and it gets the new value associated to $s_g^{A_i}$, together with the propagation attributes;
3. G-Profile verifies the existence of a mapping function on $t_h^{A_j}$ having $s_g^{A_i}$ as source object;
4. G-Profile asks the application A_i for complementary data if the target object $t_h^{A_j}$ detains a matching function needing additional data;

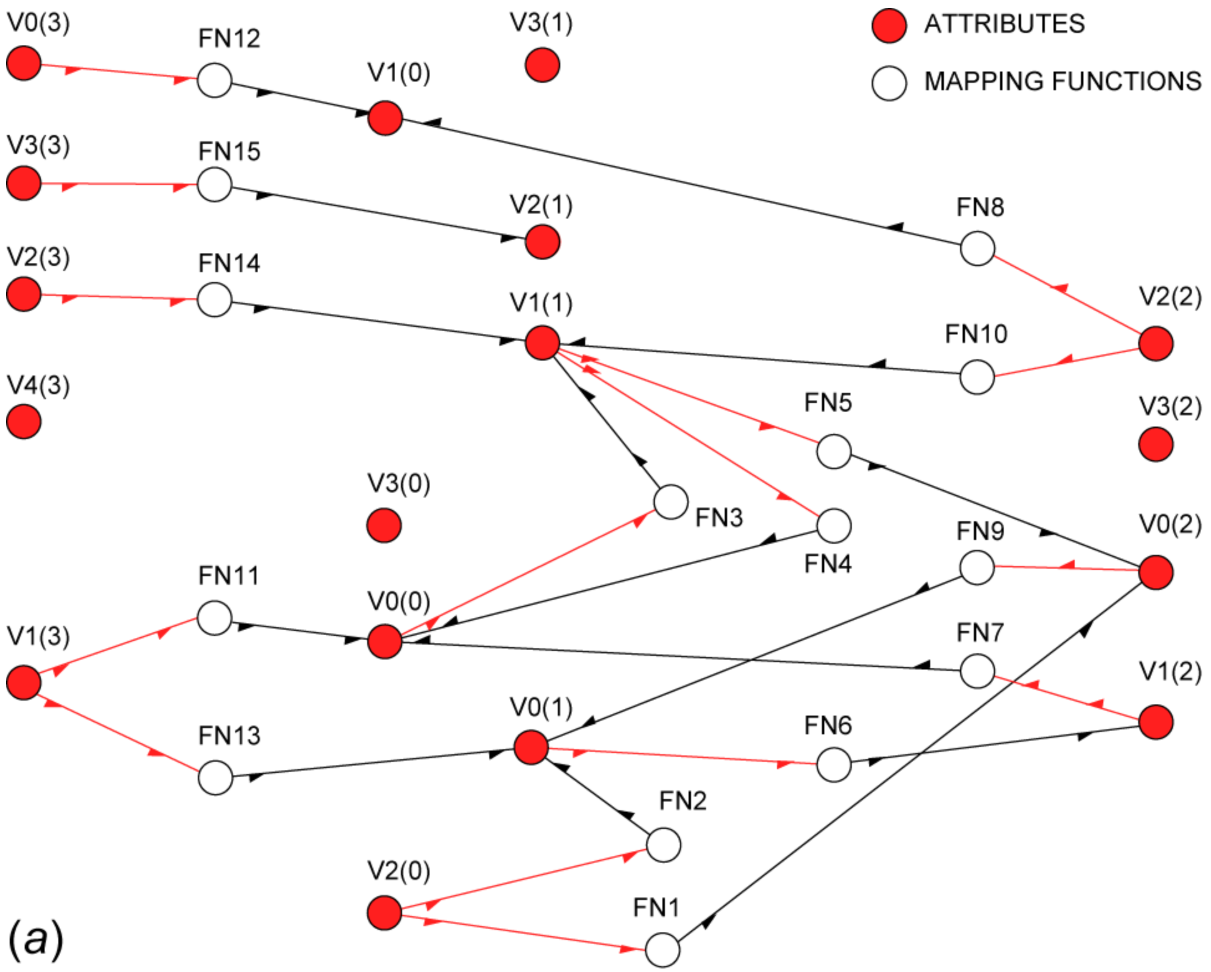
Data Propagation 2/2



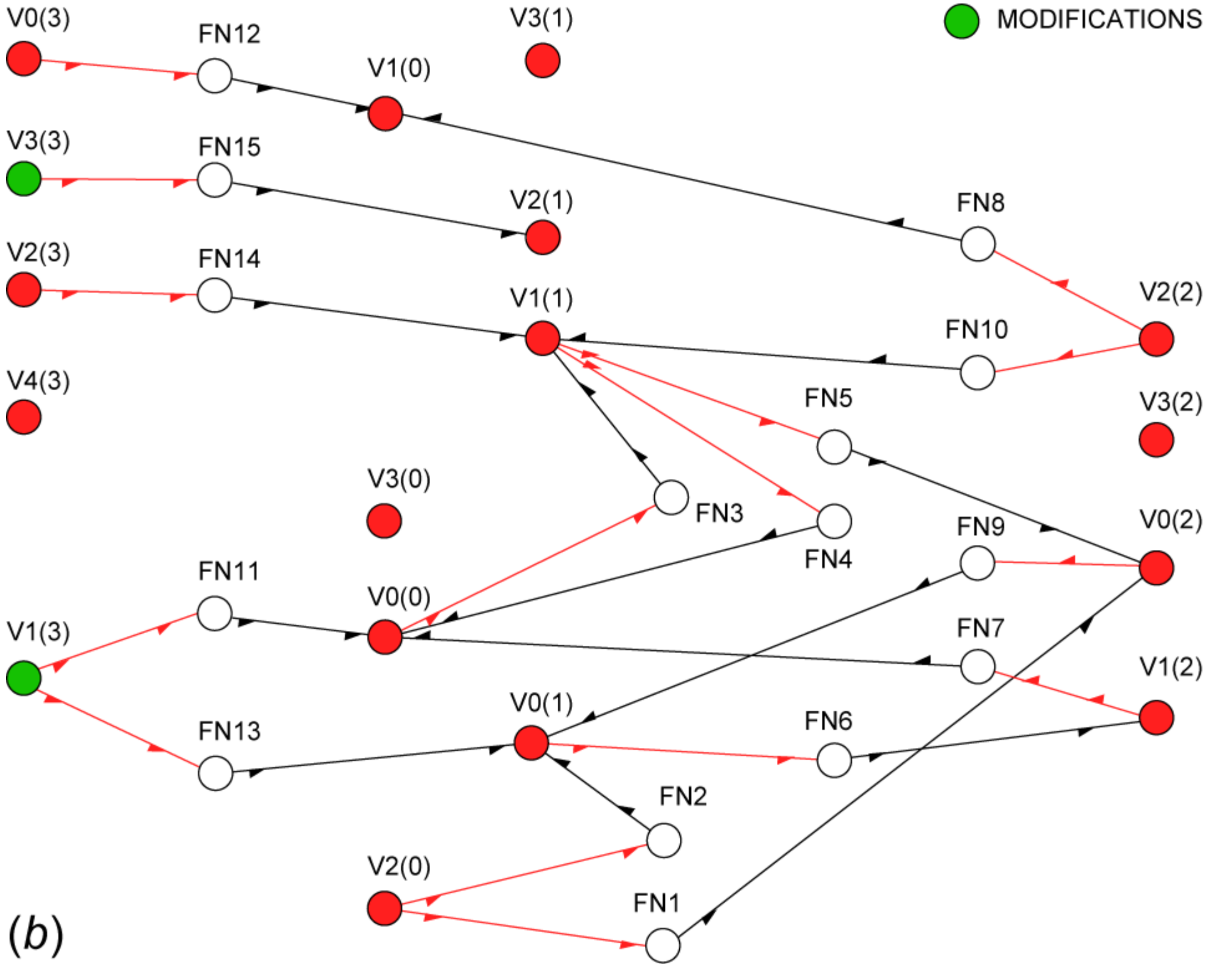
5. Once all the needed source data are available, G-Profile sends to the application A_j: (i) the modification on $s_g^{A_i}$, (ii) possible additional data necessary to the mapping function involving $s_g^{A_i}$ and $t_h^{A_j}$, (iii) the list of propagation attributes given by the application A_i;

6. Once the application A_j receives this information from G-Profile, A_j will use them in order to evaluate the conditions that will effectively permit to propagate the modification on $s_g^{A_i}$ to $t_h^{A_j}$.

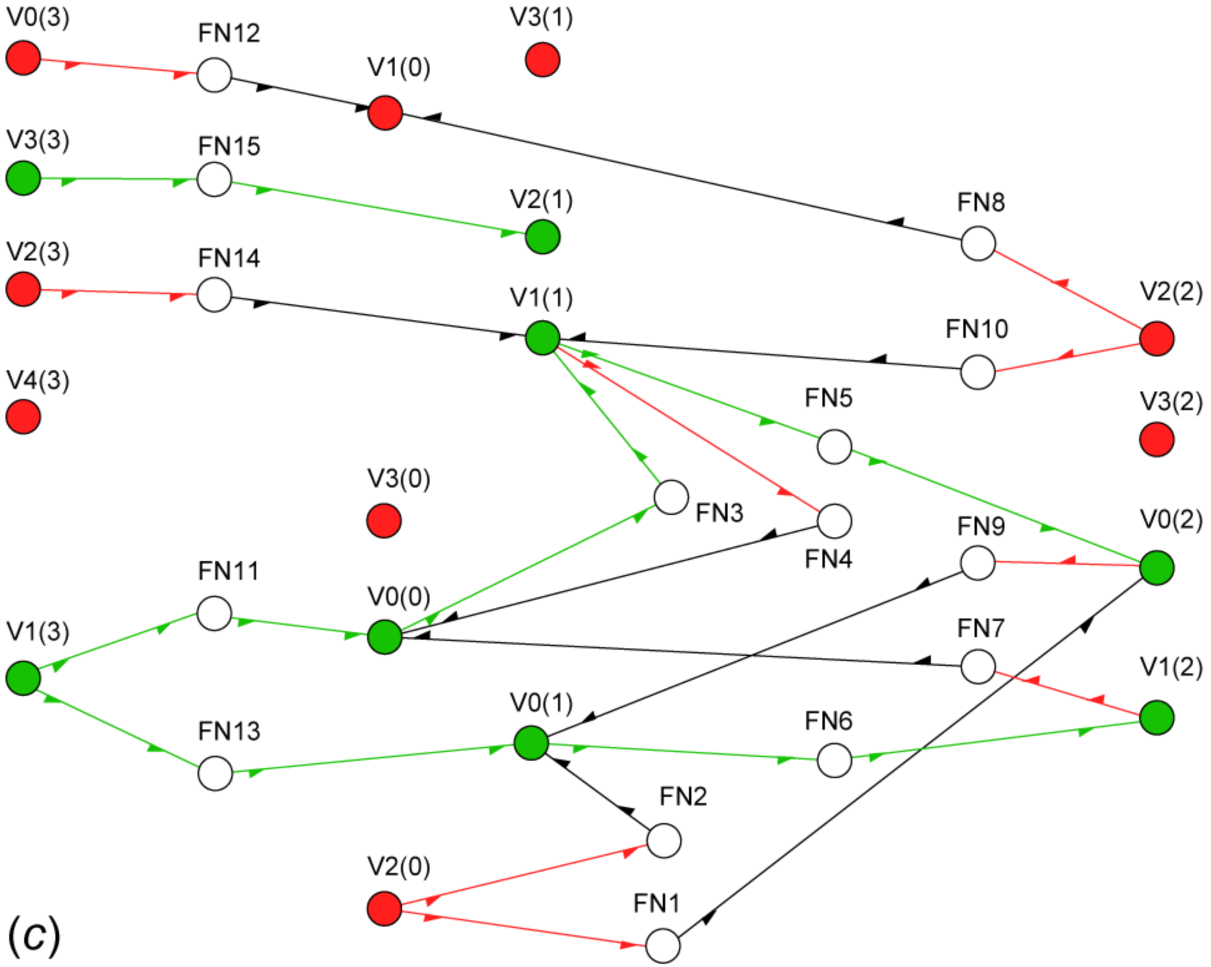
Recursive Data Propagation – 1/4



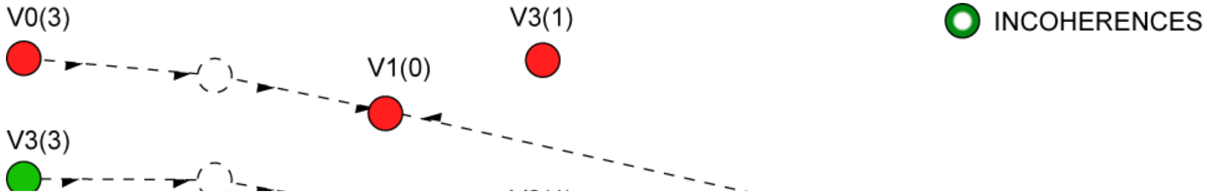
Recursive Data Propagation – 2/4



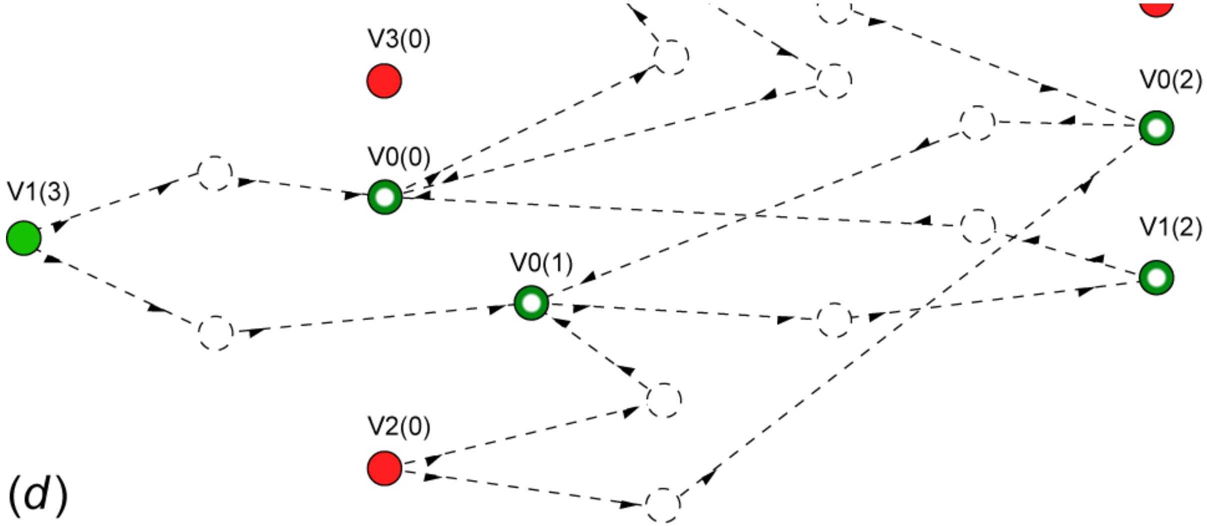
Recursive Data Propagation – 3/4



Recursive Data Propagation – 4/4



- Manual mapping creation is a time consuming process
- There are many « obvious mappings » easily identifiable

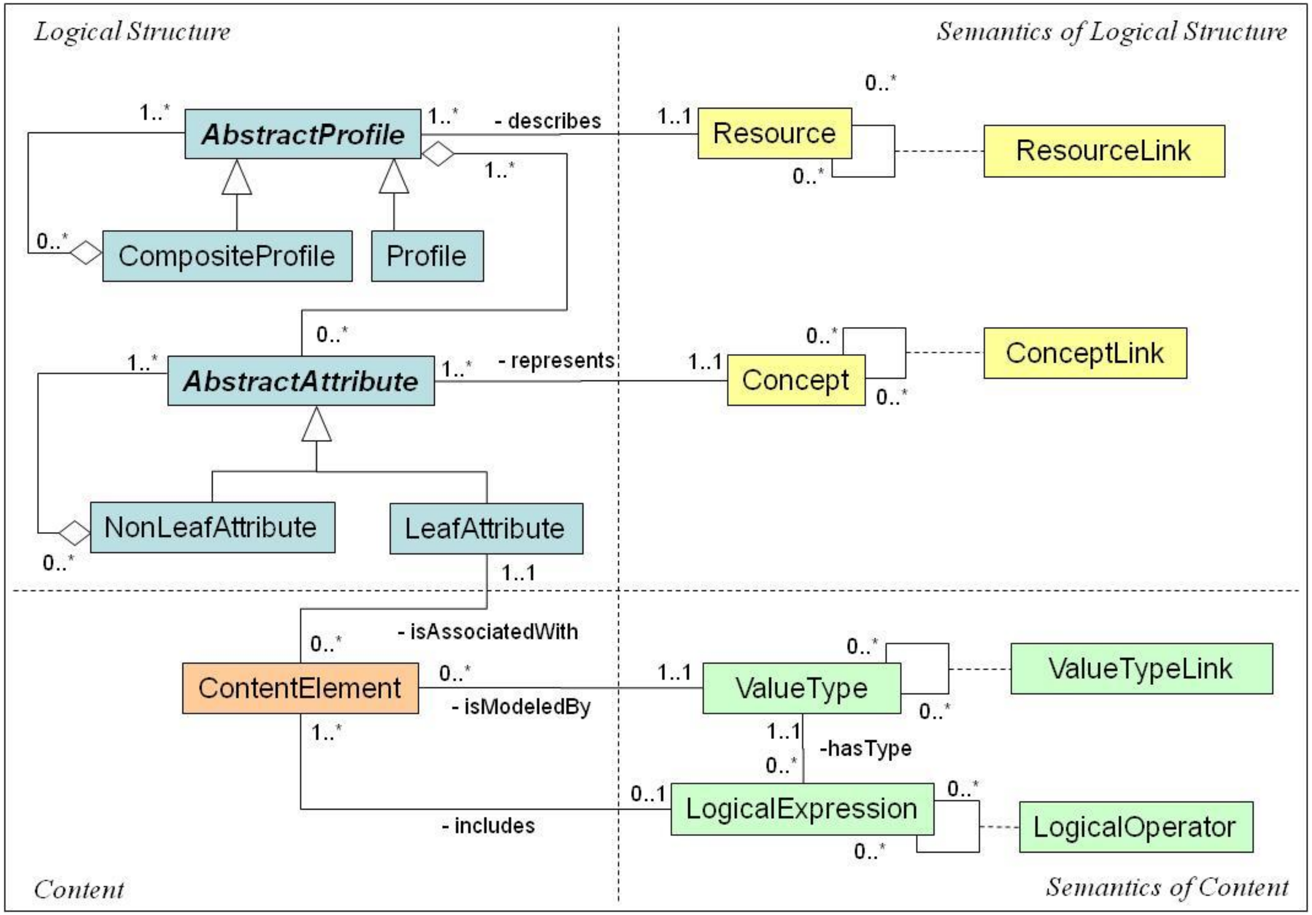


Mapping Identification

- A “semantic layer” to:
  - Allow every application to manage its own view on user profiles (e.g. different attribute names)
  - Avoid explicit description of relations between attributes
  - Identify related attributes into two user profiles coming from two applications
  - Every application uses the semantic layer to label its own attributes

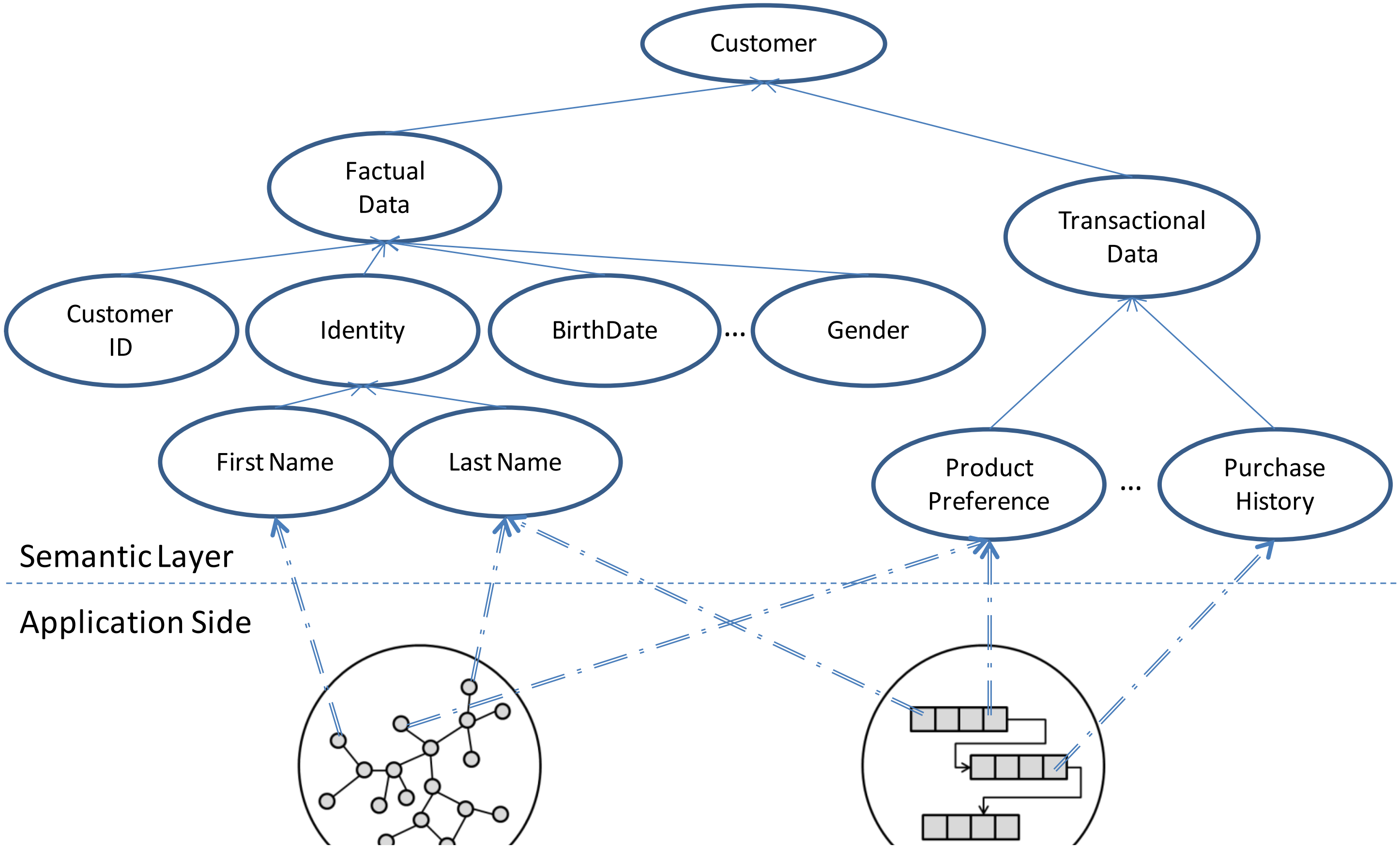
Mapping Identification

- Generalized semantic user profile



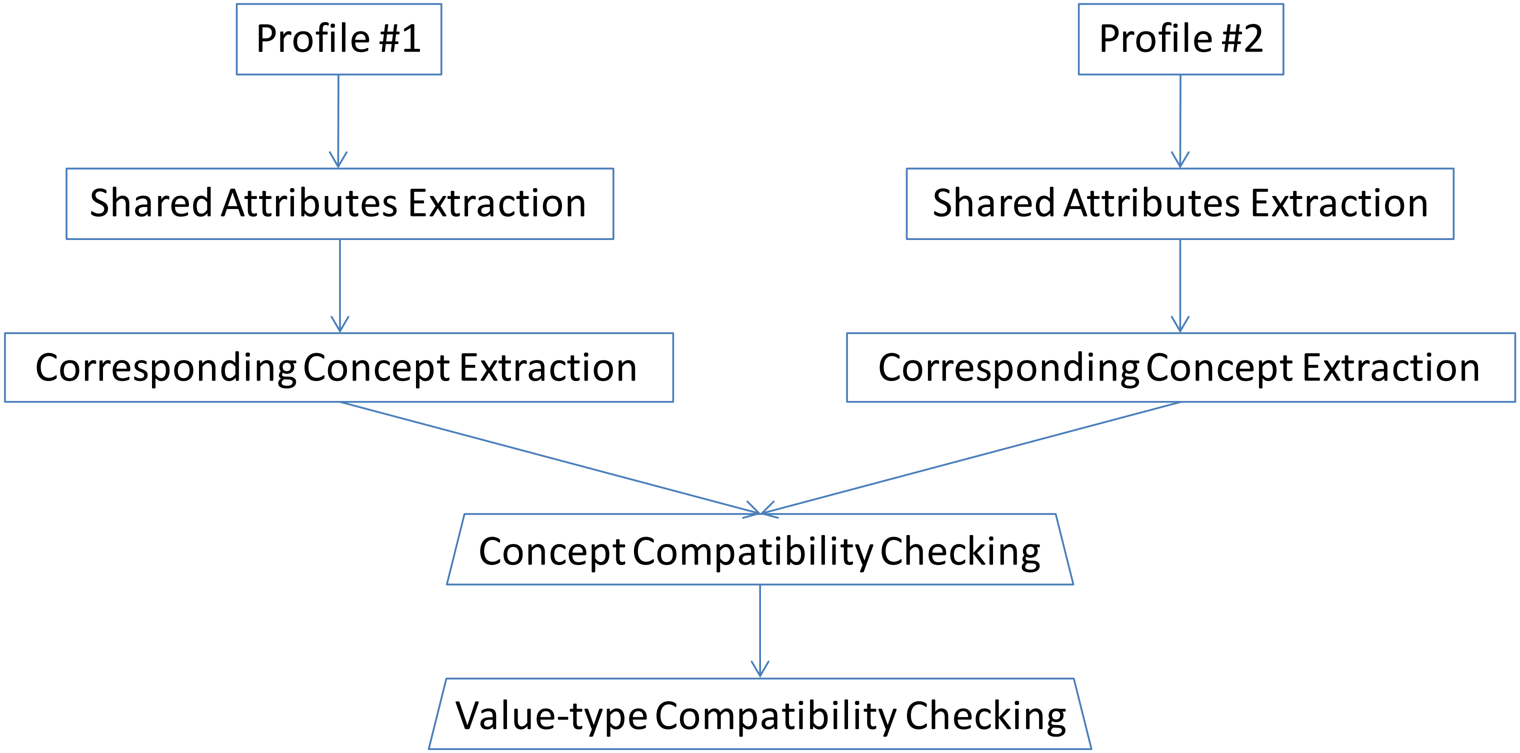
Mapping Identification

- Semantic labeling of attributes



Mapping Identification

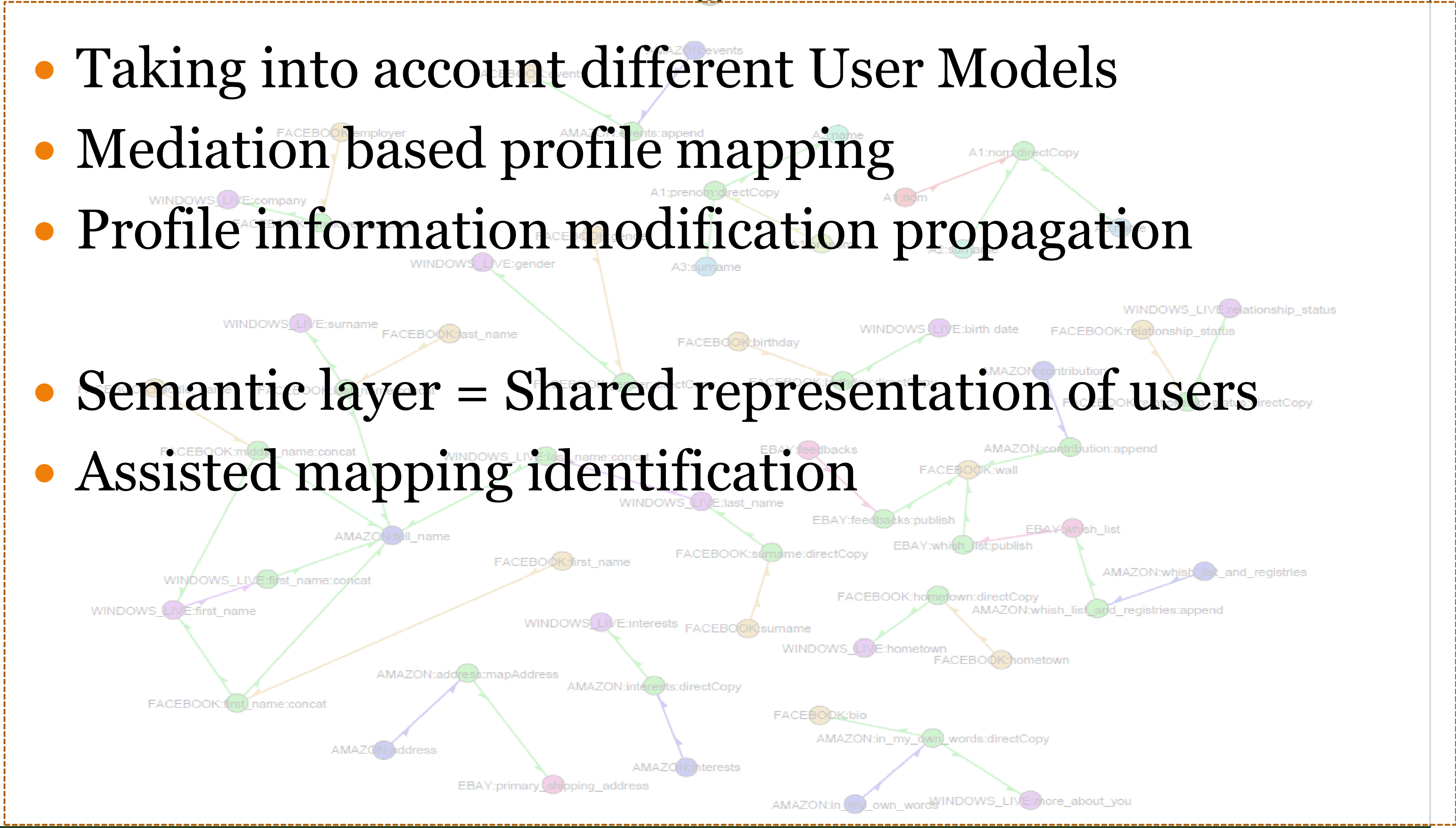
- Mapping identification process


Profile #1
Profile #2
Shared Attributes Extraction
Shared Attributes Extraction
Corresponding Concept Extraction
Corresponding Concept Extraction
Concept Compatibility Checking
Value-type Compatibility Checking

Benefits of Integration

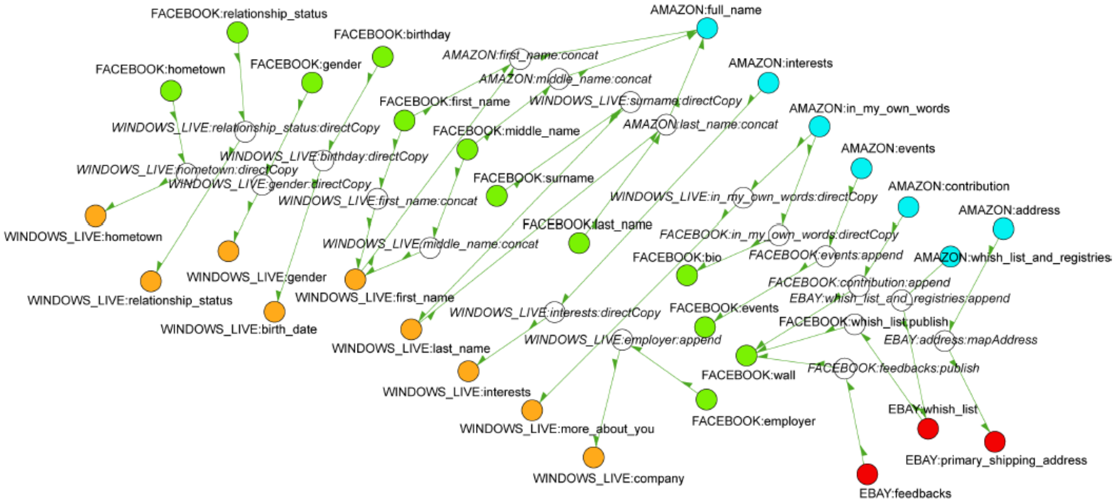
- Taking into account different User Models
- Mediation based profile mapping
- Profile information modification propagation

- Semantic layer = Shared representation of users
- Assisted mapping identification



Perspectives

- Validate the model on real or artificial data
- Handle privacy issues and refine the security and privacy issues through the semantic layer
- Integrate the semantic layer in the prototype
- Propose the model as a standard protocol



FACEBOOK:relationship_status
AMAZON:full_name
FACEBOOK:birthday
AMAZON:first_name:concat
FACEBOOK:hometown
FACEBOOK:gender
AMAZON:middle_name:concat
AMAZON:interests
FACEBOOK:first_name
WINDOWS_LIVE:surname:directCopy
AMAZON:in_my_own_words
WINDOWS_LIVE:relationship_status:directCopy
FACEBOOK:middle_name
AMAZON:last_name:concat
AMAZON:events
WINDOWS_LIVE:birthday:directCopy
WINDOWS_LIVE:hometown:directCopy
FACEBOOK:surname
WINDOWS_LIVE:gender:directCopy
AMAZON:contribution
WINDOWS_LIVE:first_name:concat
WINDOWS_LIVE:in_my_own_words:directCopy
AMAZON:address
FACEBOOK:last_name
WINDOWS_LIVE:hometown
FACEBOOK:in_my_own_words:directCopy
WINDOWS_LIVE:middle_name:concat
FACEBOOK:bio
FACEBOOK:events:append
AMAZON:whish_list_and_registries
WINDOWS_LIVE:gender
FACEBOOK:contribution:append
WINDOWS_LIVE:relationship_status
WINDOWS_LIVE:first_name
EBAY:whish_list_and_registries:append
WINDOWS_LIVE:interests:directCopy
FACEBOOK:events
WINDOWS_LIVE:birth_date
FACEBOOK:whish_list:publish
WINDOWS_LIVE:employer:append
EBAY:address:mapAddress
WINDOWS_LIVE:last_name
FACEBOOK:feedbacks:publish
FACEBOOK:wall
WINDOWS_LIVE:interests
EBAY:whish_list
WINDOWS_LIVE:more_about_you
FACEBOOK:employer
EBAY:primary_shipping_address
WINDOWS_LIVE:company
EBAY:feedbacks