Context-aware databases
design, integration and applications

Letizia Tanca
Politecnico di Milano (*)

(*) joint work with the Context-ADDICT team:
C. Bolchini, C. A. Curino, G. Orsi, E. Quintarelli, R. Rossato, F. A. Schreiber

Bertinoro, March 2008

Context-aware Data Tailoring

joint work with the Context-ADDICT team:
C. Bolchini, C. A. Curino, G. Orsi, E. Quintarelli, R. Rossato, F. A. Schreiber

Context-aware databases
design, integration and applications

Bertinoro, March 2008

The space-reduction point of view

A contribution from the database research area

CHALLENGES:

- Involved data volumes
- Data heterogeneity
- Data dynamicity
- Data distribution
- Scalability of the personalization solutions

ANSWERS:

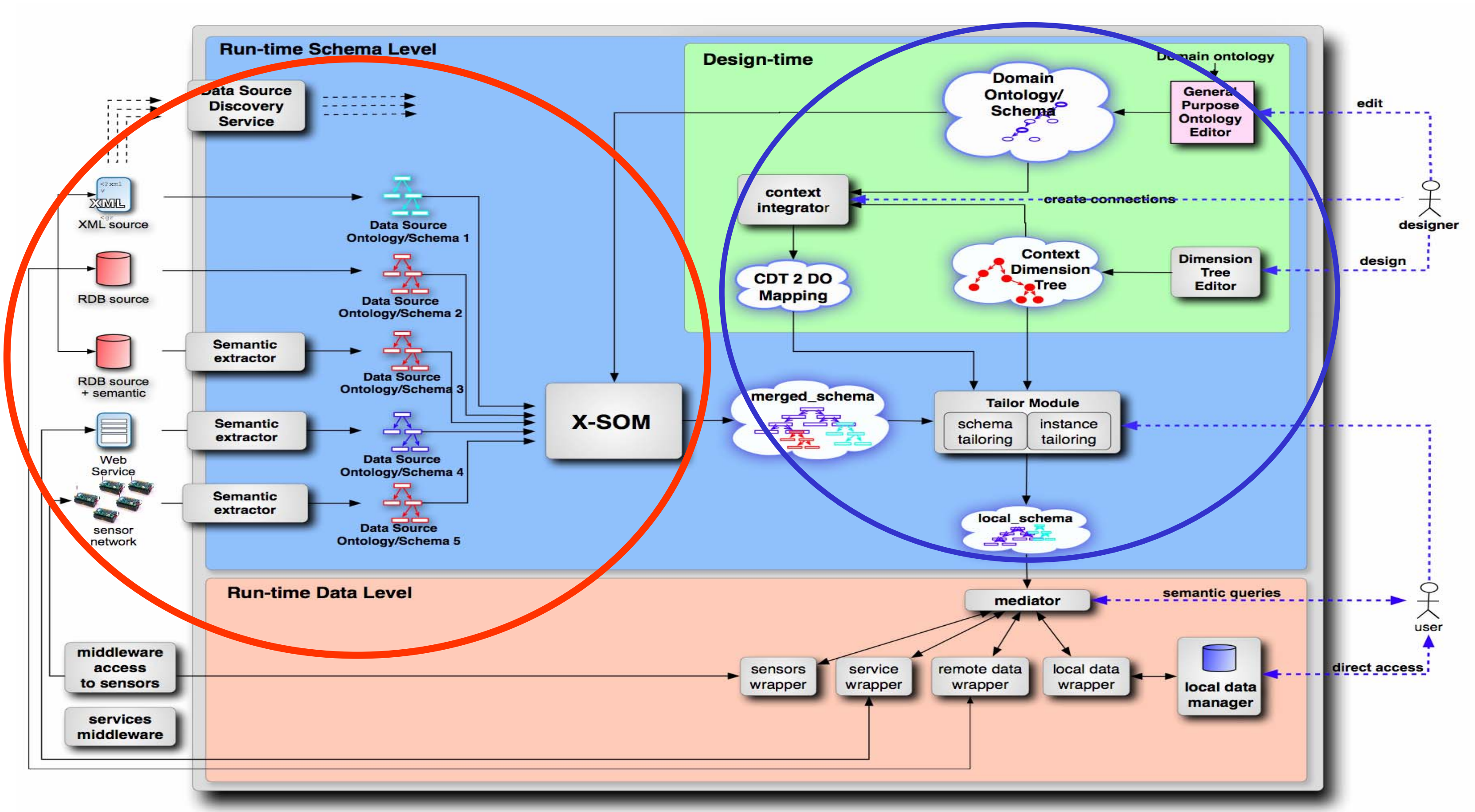
- Reduction of data volumes → context-aware data tailoring
- Data integration

Other approaches to data reduction

- **Data reduction via storage of intensional information:** **properties instead of data → data mining**
- **Statistic summarization:** **hystograms, distributions…**
- **Aggregations:** **average, sum..**
- **Data compression**
  - **All obtained by approximation: accuracy of the answer sacrifices storage space and response time**
  - **Do not solve the problem of reducing the** *information noise*

ContextADDICT ARCHITECTURE

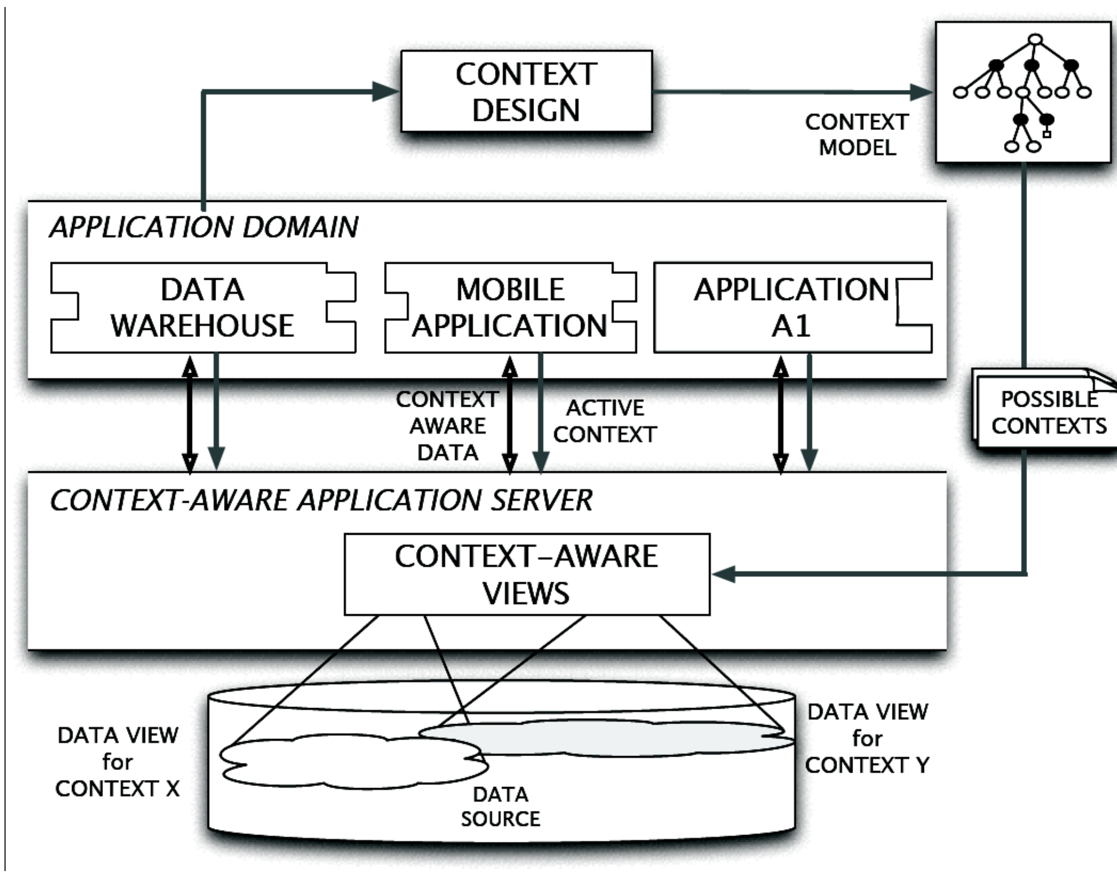
On-the-fly data integration + data reduction via tailoring



Ambient (or Context) Dimensions

- different points of view the device data are viewed from
- they **drive the portion of data to be selected,** for instance to be stored on a portable device
  - **views over the global schema**

Data tailoring architecture



The six "W" questions of context

1. What is context?
2. Who might benefit from an awareness of their context; whose context is important to whom, or what?
3. Where can an awareness of context be exploited?
4. When is context-awareness useful?
5. Why are context-aware applications useful?

Answers to these five questions underpin the higher level, meta-question of:

6. hoW do we implement context-awareness so that we can develop context-aware applications?

(Proceedings of the CHI 2000 Workshop on "The What, Who, Where, When, Why and How of Context-Awareness", David R. Morse, Anind K. Dey, 2000, Georgia Institute of Technology)

The medicare application:
a simple schema

PATIENT (SSN, FName, LName, Sex, BirthD, DeathD, Address, City, State, Zip, Phone, BloodType, Notes, MCUID, Booklet, DocID)

MEDICAL CARE UNIT (ID, Name, Address, City, State, Zip, Phone, Type)

SERVICE (ID, Name, Typology, Difficulty, Period)

USES (MCUID, SERVICEID)

PRESCRIPTION (SSN, DRUGID, Mode, Dosage,Administration, StartDate, EndDate,Comments)

DRUG (ID, Name, Posology, Ingredients,SideEffects, Manufacturer, Comments)

DRUG IN PHARMACY (DRUGID, PHARID)

PHARMACY (ID, Name, Address, City, State,Zip, Phone, OpeningHrs)

THERAPY (SSN, SERVICEID, Mode, StartDate, EndDate, Comments)

CONTEXT DIMENSIONS

INTUITIVE DIMENSIONS

- **HOLDER/ROLE**
  - TYPE (ROLE) OF USER CARRYING/USING THE DEVICE
- **INTEREST TOPIC**
  - PARTICULAR ASPECT/SUBJECT THE USER IS INTERESTED IN AT A CERTAIN MOMENT
- **SITUATION**
  - PHASE OF THE APPLICATION'S LIFE
- **INTERFACE**
  - TYPE OF ACCESS TO THE DATABASE CONTENTS
- **TIME**
  - RELATIVE OR ABSOLUTE
- **SPACE**
  - RELATIVE OR ABSOLUTE
- **DATA OWNERSHIP**
  - ACCESS RIGHTS TO THE STORED DATA

DIMENSION VALUES (1)

***holder (role)* dimension in the medical care application**

- *doctor*
  - doctors will hold information about all their patients,
- *patient*
  - patients will only hold information related to themselves, maybe at a finer level of detail.
- *each of the possible other values of this dimension (e.g. hospital administrator).*

DIMENSIONS VALUES (2)

interest topic **dimension**

- **prescriptions**
- **chronic diseases**
- **...**

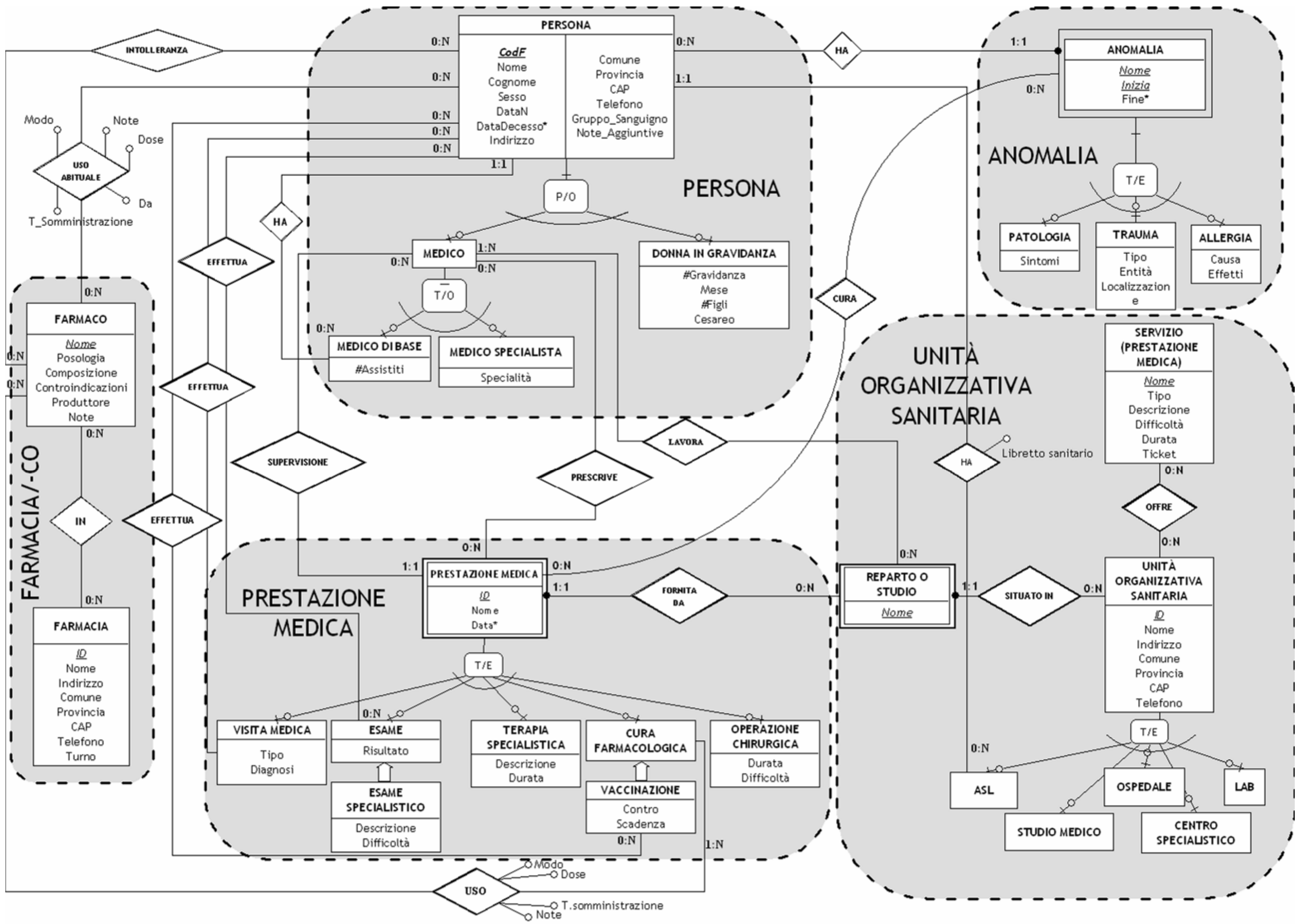
situation **dimension**

- **regular, i.e. ordinary patient's state**
- **hospitalized**
- **rehabilitation state**
- **...**

interface **dimension**

- **human**
- **machine**
- **...**

Medical Database interest topics



CONTEXT DERIVATION

Contexts are derived from the array model, e.g.:

- **<patient, chronic diseases, human, hospital,*,*>** contains all the information needed by a patient at the hospital w.r.t. his/her chronic diseases (if any).
- **<patient, prescriptions, human, regular,*,*>** contains all the information needed by a patient in a normal situation, w.r.t. his/her prescriptions (if any).
- **<doctor, prescriptions, human, regular,*,*>** contains all the information needed by a doctor regarding all his/her regular patients' prescriptions.

!! attention to the chunks' actual significance

<doctor, accounting, machine, hospital,*,*>

makes little sense in view of the application semantics. Constraints have to be designed in order to eliminate meaningless or forbidden contexts

LOGICAL CHUNK PRODUCTION

<patient,prescriptions,*,hospital,*,*>

CREATE VIEW PAT-PRESC-HOSP AS

SELECT P.SSN, P.FName, P.LName, DRUG.Name AS DrugName, Posology, SideEffects, Mode, Dosage, Administration, StartDate, EndDate, Comments, MCU.Name, MCU.Address,MCU.City, MCU.State, MCU.Zip,MCU.Phone, MCU.Type

FROM PATIENT P, DRUG, PRESCRIPTION, MEDICAL CARE UNIT MCU

WHERE P.SSN = PR.SSN AND PR.DRUGID = DRUG.ID AND P.MCUID = MCU.ID AND MCU.Type = ‘‘hospital’’

CHUNK INSTANTIATION

SELECT *

FROM *PAT-PRESC-HOSP*

WHERE *SSN = "930029747"*

AND *MCUID.NAME = ''Mt. Sinai''*

CHUNK INSTANTIATION

w.r.t. time and space:

SELECT *

FROM *PAT-PRESC-HOSP*

WHERE *SSN* = *"930029747"* AND

StartDate < *Now()* AND EndDate > *Now()*

AND City = *ThisCity()*

the *order_by clause* can be used to limit the cardinality to the *topmost n* entries

- *the most recent n* (on *time* dimension)
- *the nearest n* (on *space* dimension)
- the *cheapest n* (on *interest_topic* dimension)
- ...

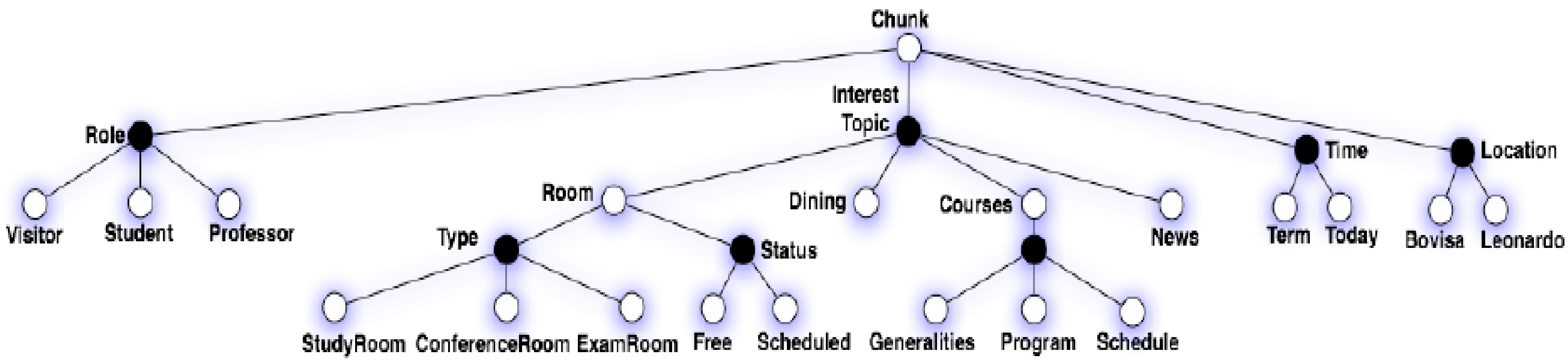
Data ownership

- Concerns read,update, delete, and insert access rights to the vsdb information, which might be different depending on the user categories
- Ownership views on data chunks
- Ownership analisys for update priorities

A more complex example: MSA

- The scenario is the *University Everyday Life (MSA)*
- Users (in this small example): Professors, Students, Visitors
- Provide context-aware data on mobile terminals (Smartphones, PDAs) and standard devices (Desktop, Laptop) about:
  - **Restaurants and bars in the area surrounding the university (each subdivision)**
  - **Free rooms (both to be reserved or just to be used)**
  - **Courses**
  - **Information about seminars and events at the Department**
  - **News about professors (schedule changes, new materials)**
  - **Data sources are heterogeneous, distributed, independent, possibly transient, possibly partially overlapping**

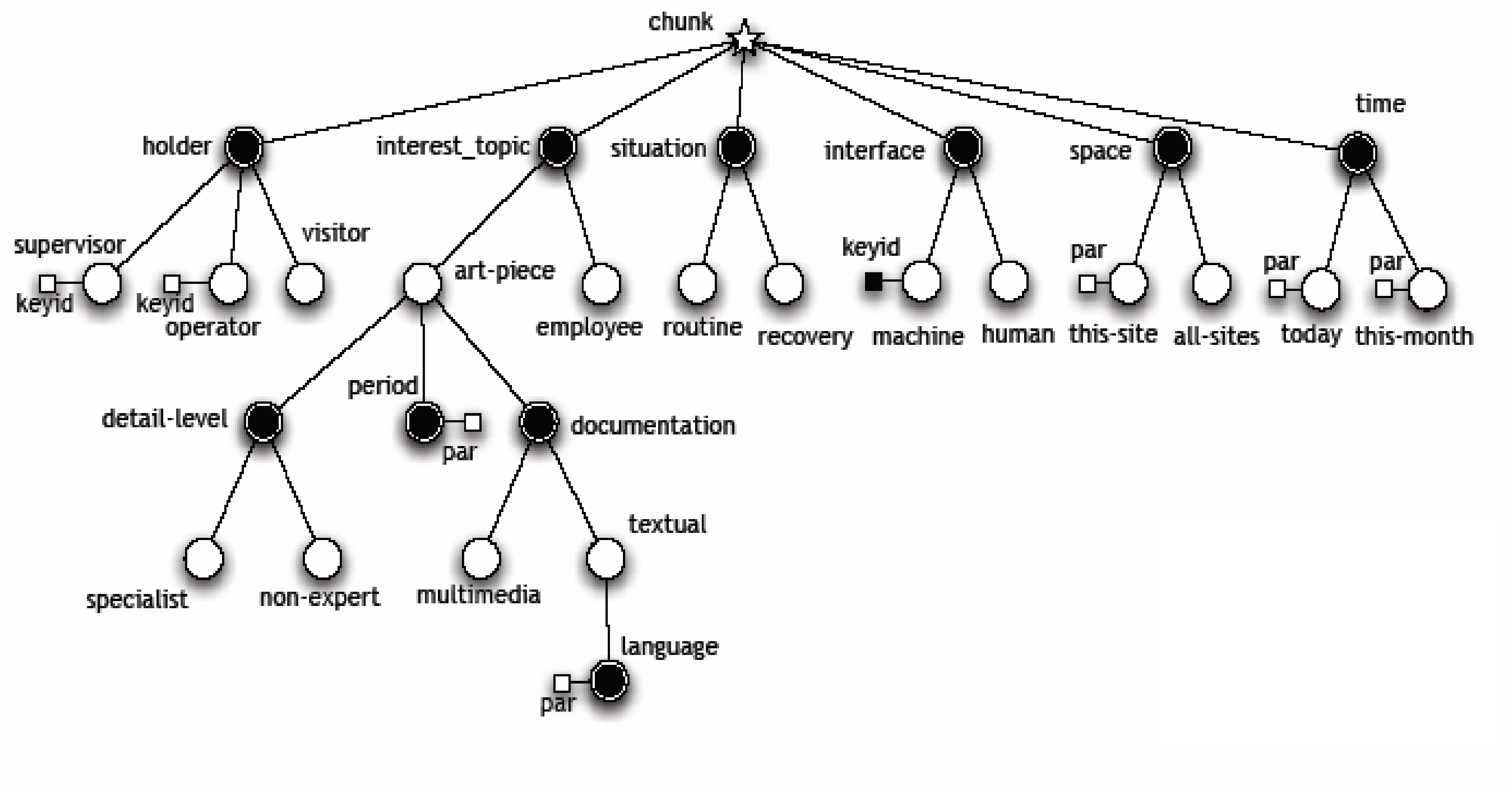
Context Modelling



Context-Dimension Tree:

- representation independent
- extensible
- granularity and (useless-context) constraints support

The archaeological site example



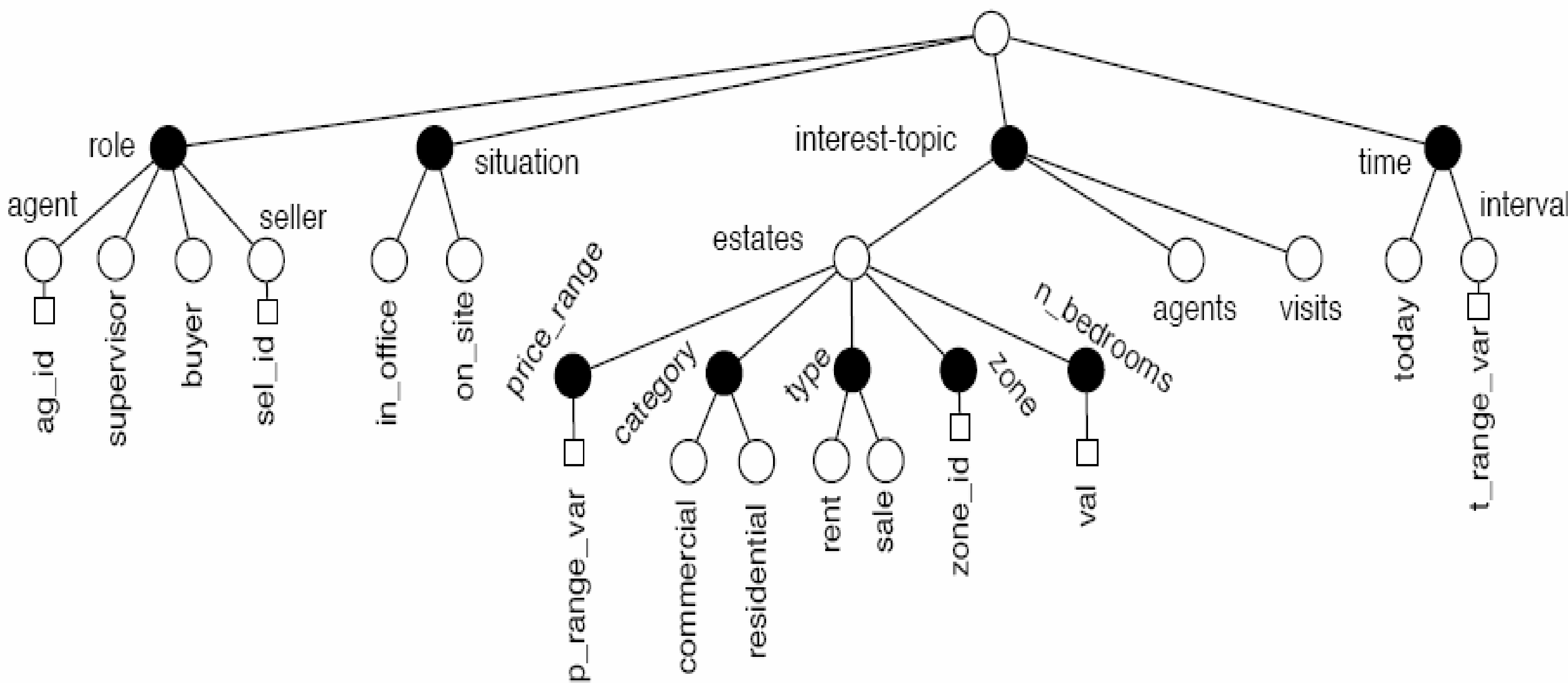
The real-estate example

OWNER(<u>IdOwner</u>, Name, Surname, Type, Address,City, PhoneNumber)
ESTATE(<u>IdEstate</u>, IdOwner, Category, Area, City, Province, RoomsNumber, Bedrooms, Garage, SquareMeters, Sheet, CadastralMap)
CUSTOMER(<u>IdCustomer</u>, Name, Surname, Type, Budget, Address, City, PhoneNum)
AGENT(<u>IdAgent</u>, Name, Surname, Office, Address,City,Phone)
AGENDA(<u>IdAgent, Data, Hour</u>, IdEstate, ClientName)
VISIT(<u>IdEstate, IdAgent, IdCustomer, Date</u>, ViewDuration)
SALE(<u>IdEstate, IdAgent, IdCustomer, Date</u>, AgreePrice, Status)
RENT(<u>IdEstate, IdAgent, IdCustomer, Date</u>, RatePrice, Status, Duration)
PICTURE(<u>IdPicture</u>, IdEstate, Date, Description, FileName)

Real-estate context dimensions

Dimension	Meaning	Examples
Role	The actors using the system.	CEO, agency manager, agent.
Interest Topic	The areas of interest for the possible users of the application.	"agencies", i.e., information about agencies (and one in particular) that can be controlled by the CEO, "agents", i.e., information about agents that can be viewed by the CEO or by the agency's manager, "customers", i.e., information about sellers and buyers that can be viewed by the agent and manager, and "properties", i.e., all the knowledge about estates to be sold or rented. This last interest topic can be further decomposed w.r.t. two different criteria: commercial/residential estates or rented/sold properties.
Situation	Phases of the application life.	The user is consulting his/her data when at the office, i.e., in_office as opposed to the on_site situation, when, for instance, an agent is showing an apartment to a prospective customer.
Time	Temporal indication based on the current time. Time can be relative or absolute and its granularity may vary.	In our example we have chosen a relative view of time, and allowed two choices: the current day, or a variable time interval centered on the current instant, suitable for data analysis.
Space	A location indication, normally referring to the place where the user is currently located. Space can be relative or absolute, and its granularity may vary.	In our example, "here" or "this city" are relative space data, while "Marina del Rey district in L.A." is absolute.
Interface	Indication of channel or presentation for delivering information.	In some application cases, some data have to be used by humans, directly perusing text and multimedia information, but in others data could be managed by electronic devices solely, requiring only compact codes.

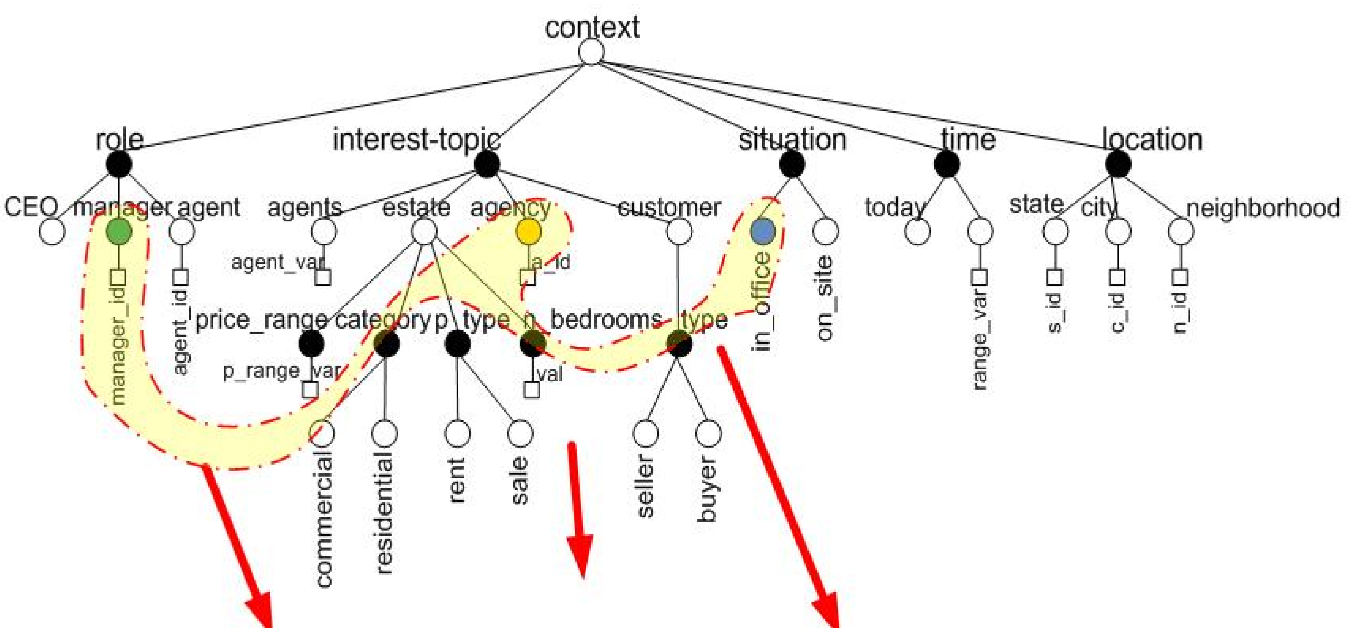
Context dimension tree of the real estate example



Real-estate context examples

Context	Corresponding data
⟨role:agent($agent_id), interest_topic:estate, situation:on_site, time:today, location:city($c_id)⟩	Agent's operational daily data: schedule estates details (including building facilities and neighborhood amenities), multimedia data compatible with his/her mobile device, agent's agenda, estate owner contact, customer contact.
⟨role:CEO, interest_topic:sales, time:month, location:city($c_id)⟩	Aggregated data about sales: e.g., average and sum of sales returns, average of commissions on sales aggregated by time and location.
⟨role:manager($manager_id), interest_topic:agents, situation:in_office, time:range($range_var)⟩	Data to assign the properties to the agents: Agents agenda, personnel and agent data, pending properties, customers requests.
⟨role:agent($agent_id), interest_topic:customers, situation:in_office, location:city($c_id)⟩	Data useful to match buyer and seller: customer's contacts, pending requests and offers details.

The agency manager, when in the office




```
CREATE VIEW ManagerAgents AS{
  SELECT P.*
  FROM Personnel AS P
  WHERE agentManager=$manager_id
}

CREATE VIEW ManagerEstates AS{
  SELECT E.*
  FROM Estate AS E, Agenda AS A
  WHERE E.estateID = A.estateID AND
        A.agentID IN (SELECT personellID
                      FROM ManagerAgents)
}

CREATE VIEW ManagerSale AS{
  SELECT S.*
  FROM Sale AS S
  WHERE S.agentID IN (SELECT personellID
                      FROM ManagerAgents)
}

CREATE VIEW ManagerRent AS{
  SELECT R.*
  FROM Rent AS R
  WHERE R.agentID IN (SELECT personellID
                      FROM ManagerAgents)
}
```

Terminology and notation

$$\mathcal{T} = \langle N, E, r \rangle$$

$$N = N_D \cup N_C$$ *dimensions (black), concepts (white)*

- The root is a *concept node*
- The root's children are the *top dimensions*
- Each generation contains *nodes of the same color*

Formal definition

- Colors are *alternated* while descending the tree:

$$\forall e = \langle n, m \rangle \in E, \quad \text{either } n \in N_D \wedge m \in N_C \text{ or } n \in N_C \wedge m \in N_D;$$

- It is possible to add a parameter to concept (white) nodes and leaf dimension (black) nodes.
- White node *parameters* indicate how to select a specific set of data instances
  - e.g. the **agent_id** for the **agent** role
- Leaf black node *parameters* indicate a selection parameter whose instances represent the possible values of that (sub-) dimension
  - e.g. the **price_range_var** for the dimension **price range**
- Dimension nodes without concept children *must have* a parameter
- Dimension nodes with concept children *do not* have a parameter

Formal definition

- **A context is formalized as a conjunction of propositions expressing context elements, each of them of the form**

`dim name = value`

- **`value`** can be
  - a concept node or
  - a concept node filtered by the value of its parameter or
  - the value for a parameter of a sub-dimension (black) leaf node
- **The values for each context element may be at any level in the tree, thus allowing for different levels of granularity.**

Formal definition

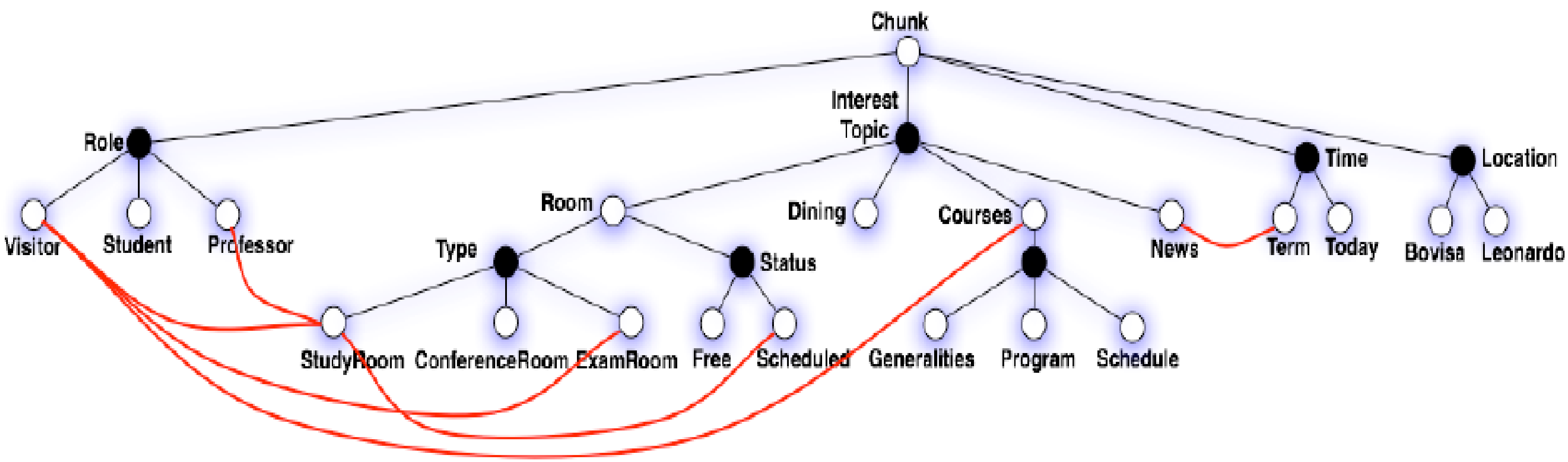
- **Sibling white nodes are mutually exclusive in a context, since they represent orthogonal concepts.**
- **When building a context, for each top dimension, at each level,** *only one white node* **among each set of siblings, and** *any number of black siblings* **may be included.**
- **A context can lack some dimension value(s): this means that those dimensions are not taken into account to tailor data, i.e., the view corresponding to that context** *does not filter the data for these dimension(s).*

Example

- Let us consider the situation of an agent (whose id. is 23564), ready to take prospective buyers to visit the residential estate properties located in the "Piola" area ($zone id="Piola")
- The current context C is the conjunctive propositional formula:

```
role = agent("23564")
^ category = residential
^ type = sale
^ zone = "Piola"
^ situation = on_site
^ time = today(getdate())
```

Context Modeling



- **Not all configurations make sense: e.g., there is no point in combining *the visitor's* role with the *courses* data**
- *Constraints* **are specified over the tree by means of a standard logical formula**
- **The most common constraints are the *forbid* (or "useless-context") constraints**

Constraints

- The designer can express constraints or preferences on the possible combinations of the context elements
- When we combinatorially generate the complete set of contexts, many contexts get discarded
- Constraints are expressed by means of formulae of propositional logic:

$$\neg(\bigwedge \texttt{context\_element\_proposition})$$

where a `context element proposition` is:

- a proposition of the form `dim name=value` or
- a disjunction of such propositions.

Interesting constraints

- *useless context (forbid)* **constraints allow the context designer to specify configurations that are not significant, thus discarding those that would represent semantically meaningless context situations or would be irrelevant for the application. In the MSA:**

$$\neg (role = visitor$$

$$\wedge\ (Desc(type = courses)\ \vee\ type = courses\ )$$

Interesting constraints

- *dimension independent constraints* **are used when the values of a (sub-) dimension do not influence the views associated with contexts containing some values for other dimensions. In the agency example:**

¬ (role = supervisor ^

^ (situation = in_office v situation = on_site))

The data associated with the supervisor's role are independent of the situation

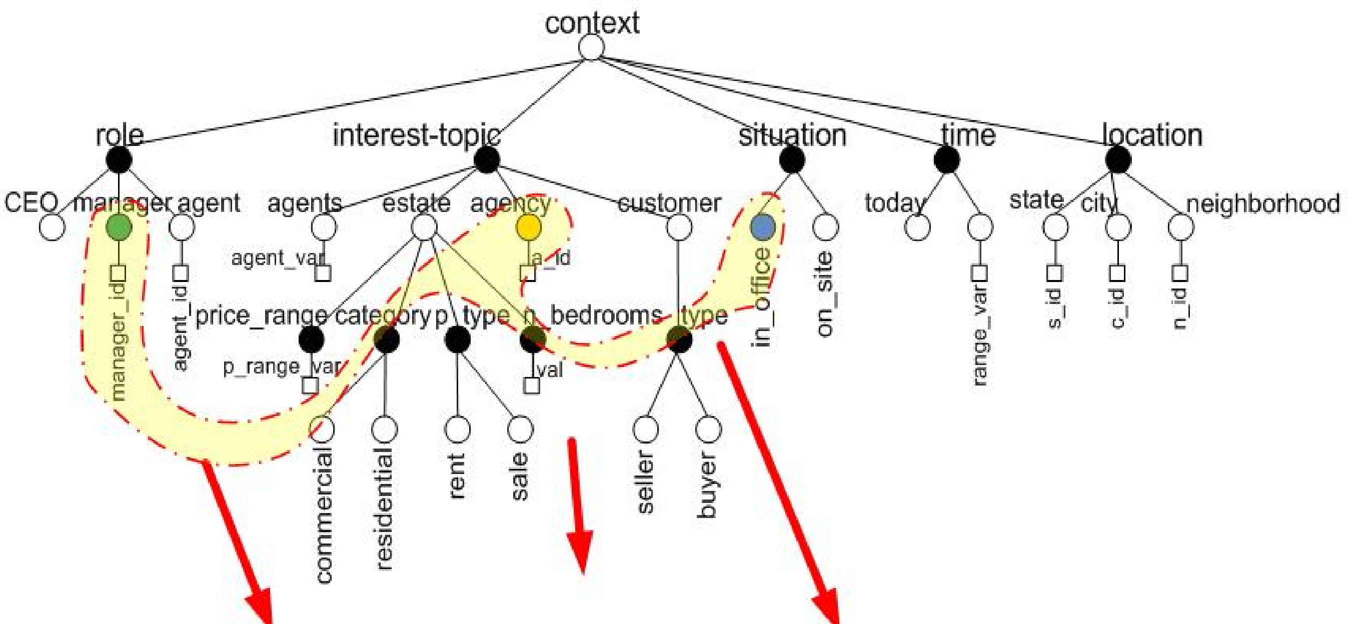
Interesting constraints

- *preferred-detail* **constraints allow the designer to express the level of detail to be preferred for a dimension, considering the other dimension values. In the agency example:**

```
¬ (role = supervisor
^ Desc(interest-topic = estates))
```

The supervisor has access to all the data related to estates, other roles may be interested in lower level details

Recall what we want to do




```
CREATE VIEW ManagerAgents AS{
  SELECT P.*
  FROM Personnel AS P
  WHERE agentManager=$manager_id
}

CREATE VIEW ManagerEstates AS{
  SELECT E.*
  FROM Estate AS E, Agenda AS A
  WHERE E.estateID = A.estateID AND
        A.agentID IN (SELECT personellID
                      FROM ManagerAgents)
}
```

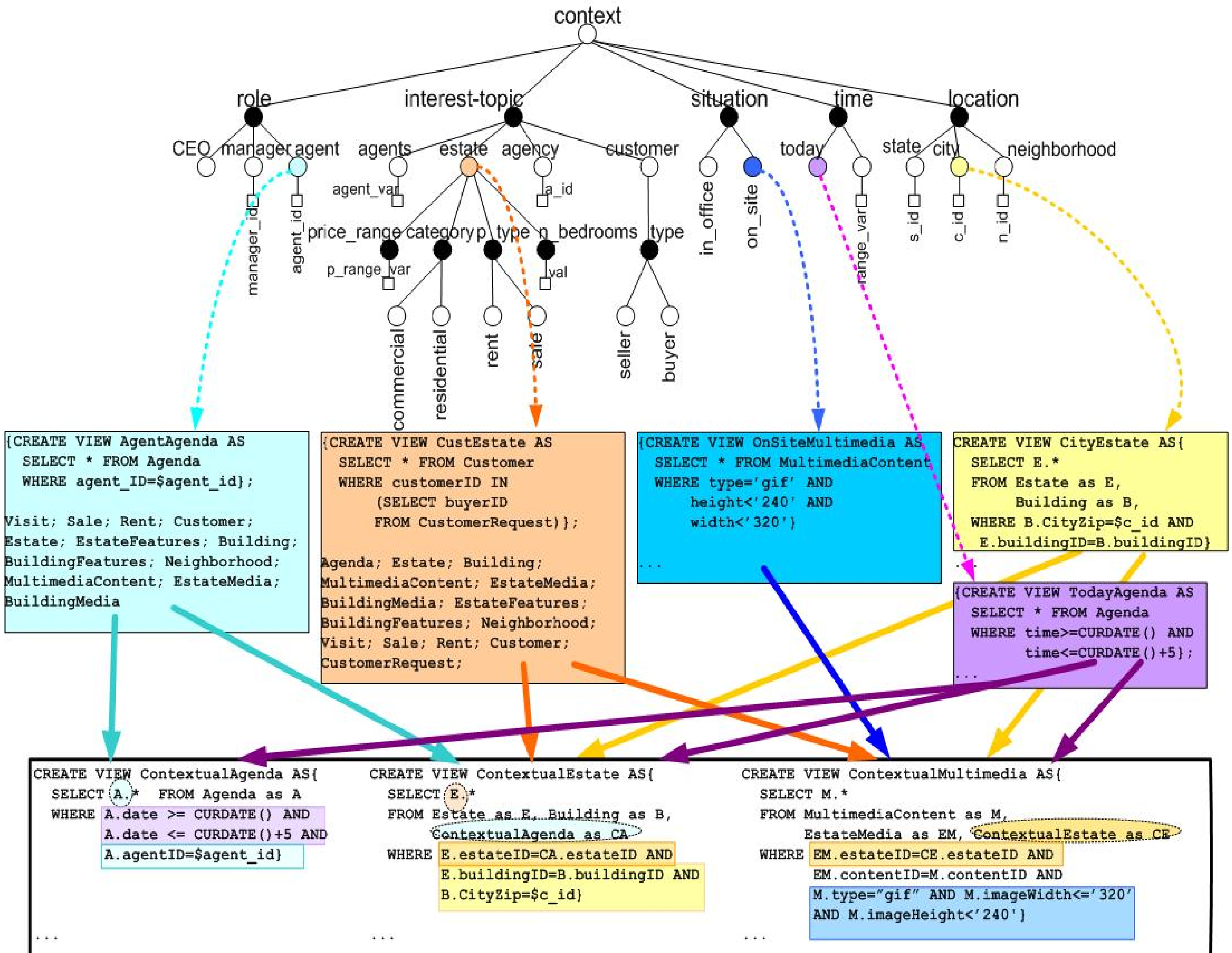
```
CREATE VIEW ManagerSale AS{
  SELECT S.*
  FROM Sale AS S
  WHERE S.agentID IN (SELECT personellID
                      FROM ManagerAgents)
}

CREATE VIEW ManagerRent AS{
  SELECT R.*
  FROM Rent AS R
  WHERE R.agentID IN (SELECT personellID
                      FROM ManagerAgents)
}
```

Obtaining relevant areas

- The relevant area, or view, related to a context C, is denoted by $\mathcal{R}el(\mathrm{C})$
- Assigning relevant areas to all possible (valid) contexts of a tree is a very time-consuming task
- Less time-consuming, but more difficult from a conceptual viewpoint, is deriving the context view from the composition of relevant areas of the component nodes
- Integration operators

Obtaining relevant areas



Context-view assignment: INTEGRATION OPERATORS

Let $\mathcal{A}$ and $\mathcal{B}$ be sets of relations.

- **Double-union:**

$$\mathcal{A} \Cup \mathcal{B} = \left\{ R \;\middle|\; \begin{array}{l} (R \in \mathcal{A} \wedge \neg\exists R' \in \mathcal{B},\ \mathtt{Sch}(R) \cap \mathtt{Sch}(R') \neq \emptyset) \ \vee \\ (R \in \mathcal{B} \wedge \neg\exists R' \in \mathcal{A},\ \mathtt{Sch}(R) \cap \mathtt{Sch}(R') \neq \emptyset) \ \vee \\ (R = \pi_S(R_A) \cap \pi_S(R_B),\ R_A \in \mathcal{A},\ R_B \in \mathcal{B}, S = \mathtt{Sch}(R_A) \cap \mathtt{Sch}(R_B) \neq \emptyset) \end{array} \right\}$$

- **Double-intersection:**

$$\mathcal{S} = \{X \,|\, X = \pi_S(X_A) \cap \pi_S(X_B)\ s.t.\ X_A \in \mathcal{A},\ X_B \in \mathcal{B}, S = \mathtt{Sch}(X_A) \cap \mathtt{Sch}(X_B) \neq \emptyset)\}$$

$$\mathcal{A} \Cap \mathcal{B} = \left\{ R \;\middle|\; \begin{array}{l} (R = R_1 \cup R_2,\ \forall R_1, R_2 \in (\mathcal{S} \cup \mathcal{A} \Cap \mathcal{B})\ s.t.\ \mathtt{Sch}(R_1) = \mathtt{Sch}(R_2)) \vee \\ (R \in \mathcal{S} \wedge\ \nexists R' \in \mathcal{S}\ s.t.\ \mathtt{Sch}(R) = \mathtt{Sch}(R')) \end{array} \right\}$$

Operator properties

- **Both operators are** *commutative* **and** *associative*
- *Selection-preserving:*

Let $\mathcal{A}$ *and* $\mathcal{B}$ *be sets of relations, and let*

$R_A = \Pi_{Z_A}(\sigma_{\theta_A}(R)) \in \mathcal{A}$ *and* $R_B = \Pi_{Z_B}(\sigma_{\theta_B}(R)) \in \mathcal{B}$

If $Z = Z_A \cap Z_B$ *then*

$$Y = \pi_Z(\sigma_{\theta_A \wedge \theta_B}(R)) \in \mathcal{A} \Cup \mathcal{B}$$

with Z non empty , and

$$Y = \pi_Z(\sigma_{\theta_A \vee \theta_B}(R)) \in \mathcal{A} \Cap \mathcal{B}$$

Relevant area assigment

The designer is in charge of assigning to each context element

$$(c \in N_C \cup \overline{N_D})$$

a *relevant area*, **or** *partial view:*

$$\mathcal{R}el : N_C \cup \overline{N_D} \rightarrow \wp(\mathcal{V})$$

According to two different policies:

- *Maximal* **relevant area**
- *Minimal* **relevant area**

Maximal relevant area

The partial view for each context element contains all information that might be interesting for that element.

OWNER(<u>IdOwner</u>, Name, Surname, Type, Address,City, PhoneNumber)
ESTATE(<u>IdEstate</u>, IdOwner, Category, Area, City, Province, RoomsNumber, Bedrooms, Garage, SquareMeters, Sheet, CadastralMap)
CUSTOMER(<u>IdCustomer</u>, Name, Surname, Type, Budget, Address, City, PhoneNum)
AGENT(<u>IdAgent</u>, Name, Surname, Office, Address,City,Phone)
AGENDA(<u>IdAgent, Data, Hour</u>, IdEstate, ClientName)
VISIT(<u>IdEstate, IdAgent, IdCustomer, Date</u>, ViewDuration)
SALE(<u>IdEstate, IdAgent, IdCustomer, Date</u>, AgreePrice, Status)
RENT(<u>IdEstate, IdAgent, IdCustomer, Date</u>, RatePrice, Status, Duration)
PICTURE(<u>IdPicture</u>, IdEstate, Date, Description, FileName)

$$\mathcal{R}el(\text{estate}) = \{\text{ESTATE}, \text{OWNER}, \text{VISIT}, \text{SALES}, \text{RENT}, \text{AGENDA}, \text{PICTURE}\}$$

Minimal relevant area

The partial view for each context element contains only information that is strictly related to that element.

OWNER(<u>IdOwner</u>, Name, Surname, Type, Address,City, PhoneNumber)
ESTATE(<u>IdEstate</u>, IdOwner, Category, Area, City, Province, RoomsNumber, Bedrooms, Garage, SquareMeters, Sheet, CadastralMap)
CUSTOMER(<u>IdCustomer</u>, Name, Surname, Type, Budget, Address, City, PhoneNum)
AGENT(<u>IdAgent</u>, Name, Surname, Office, Address,City,Phone)
AGENDA(<u>IdAgent, Data, Hour</u>, IdEstate, ClientName)
VISIT(<u>IdEstate, IdAgent, IdCustomer, Date</u>, ViewDuration)
SALE(<u>IdEstate, IdAgent, IdCustomer, Date</u>, AgreePrice, Status)
RENT(<u>IdEstate, IdAgent, IdCustomer, Date</u>, RatePrice, Status, Duration)
PICTURE(<u>IdPicture</u>, IdEstate, Date, Description, FileName)

$$\mathcal{R}el(\mathtt{estate}) = \{\text{ESTATE}, \text{PICTURE}\}$$

Relevant area assigment

- **The designer is in charge of assigning to each context element its *relevant area*, or *partial view***
- **Such attribution has to be made in a *coherent fashion***
- *Let*

$\mathcal{R}el(w) \subseteq \mathcal{R}el(k)$ **iff** $\forall R_i \in \mathcal{R}el(w) \exists R_j \in \mathcal{R}el(k)$ s.t. $\mathbf{info}_{\mathbf{t}}(R_i) \subseteq \mathbf{info}_{\mathbf{t}}(R_j)$

Assumption:

For each pair of context elements n and m,

if $n \prec m$ then $\mathcal{R}el(n) \supseteq \mathcal{R}el(m)$

Where $n \prec m$ means that *n* is more abstract than *m*,

i.e., *m* is a descendant of *n*

Example

$$\mathcal{R}el(\texttt{estate}) = \{\textsc{Estate}, \textsc{Owner}, \textsc{Visit}, \textsc{Sales}, \textsc{Rent}, \textsc{Agenda}, \textsc{Picture}\}$$

Consider now the estates belonging to the “residential” category :

$$\mathcal{R}el(\texttt{residential}) = \{\sigma_{Category=\text{“}Residential\text{”}}\textsc{Estate}, \textsc{Owner} \ltimes (\sigma_{Category=\text{“}Residential\text{”}}\textsc{Estate}),$$
$$\textsc{Visit} \ltimes (\sigma_{Category=\text{“}Residential\text{”}}\textsc{Estate}), \textsc{Sale} \ltimes (\sigma_{Category=\text{“}Residential\text{”}}\textsc{Estate}),$$
$$\textsc{Rent} \ltimes (\sigma_{Category=\text{“}Residential\text{”}}\textsc{Estate}), \textsc{Agenda} \ltimes (\sigma_{Category=\text{“}Residential\text{”}}\textsc{Estate}),$$
$$\textsc{Picture} \ltimes (\sigma_{Category=\text{“}Residential\text{”}}\textsc{Estate})\}$$

Relevant area assigment

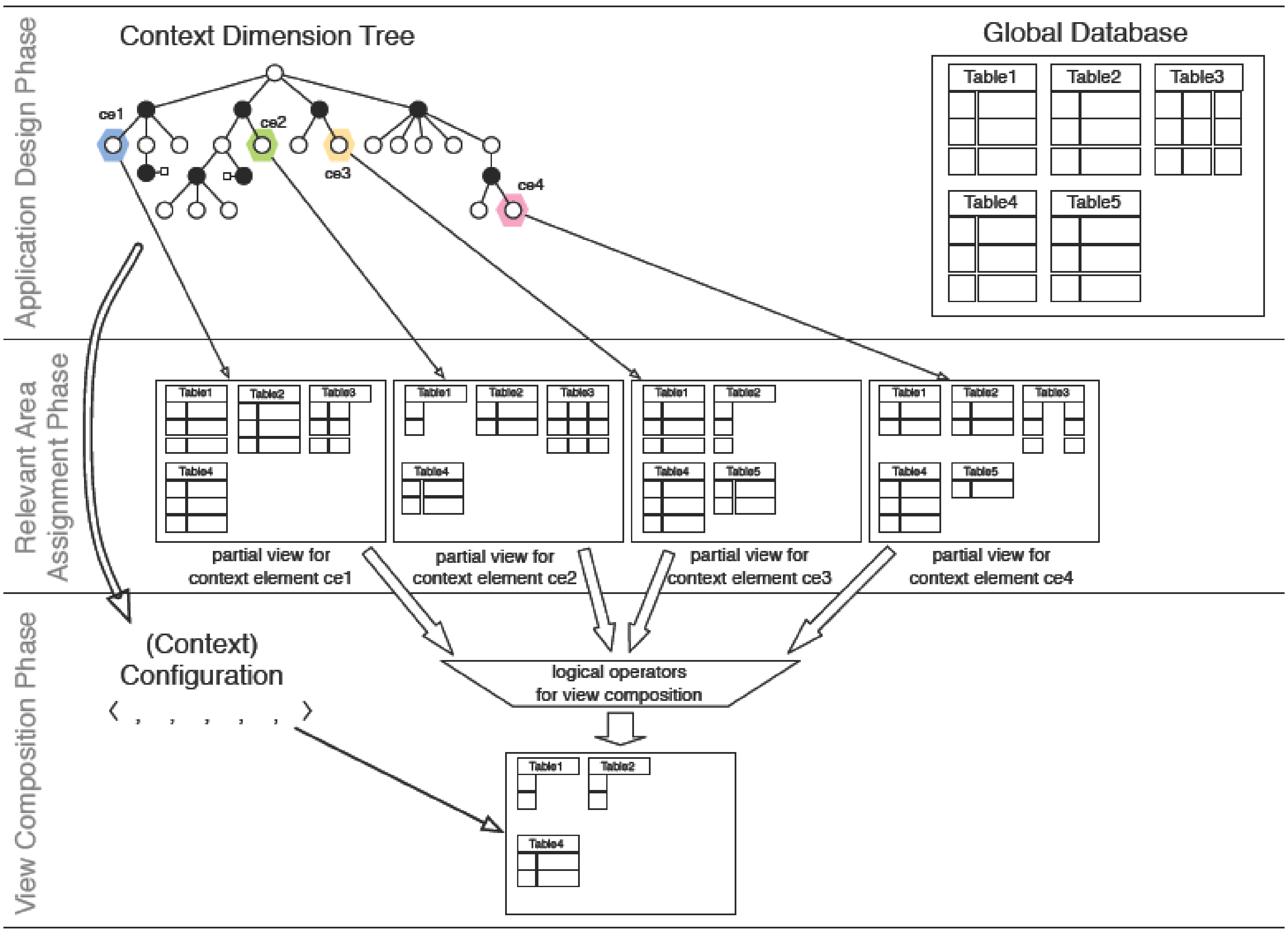
Maximal area policy:

- **navigating the CDT** *top-down*
- **Specifying, for each node** ***m***, **a** *partial view* $(\mathcal{R}el(m))$ **over the (previously defined) view** $(\mathcal{R}el(n))$ **pertaining to** ***n***, **with** $n \prec m$

Minimal area policy:

- **navigating the CDT** *bottom-up*
- **Composing the partial view of** ***n*** **(e.g. by means of the double union operator) from the partial views of its children**
- **Important: when assigning relevant areas, keep the keys!!!**

Context refinement via view composition



Example (minimal area policy)

Let:

$$\mathcal{R}el(\texttt{buyer}) = \{\Pi_Z \text{ESTATE}, \text{PICTURE}\}$$

with:

$$Z = \{IdEstate, Category, Area, City, Province, RoomsNumber, Bedrooms, Garage, SquareMeters\}$$

and:

$$\mathcal{R}el(\texttt{residential}) = \{\sigma_{Category=\text{``}Residential\text{''}}\text{ESTATE}, \text{PICTURE} \ltimes (\sigma_{Category=\text{``}Residential\text{''}}\text{ESTATE})\}$$

Then for: $C = \langle \texttt{role} : \texttt{buyer}, \texttt{category} : \texttt{residential} \rangle$

we can apply double union, and have:

$$\mathcal{R}el(\texttt{C}) = \mathcal{R}el(\texttt{buyer}) \Cup \mathcal{R}el(\texttt{residential})$$
$$= \{\sigma_{Category=\text{``}Residential\text{''}}(\Pi_Z(\text{ESTATE})), (\text{PICTURE}) \ltimes \sigma_{Category=\text{``}Residential\text{''}}((\text{ESTATE}))\}$$

Example (maximal area policy)

Let:

$$\mathcal{R}el(\texttt{agent}(\$\texttt{ag\_id})) = \{\sigma_{IdAgent=\$ag\_id}\textsc{Agent}, \sigma_{IdAgent=\$ag\_id}\textsc{Agenda}, \textsc{Visit}, \textsc{Sales}, \textsc{Rent}, \textsc{Owner}, \textsc{Customer}, \textsc{Estate}, \textsc{Picture}\}$$

and:

$$\mathcal{R}el(\texttt{residential}) = \{\sigma_{Category=\text{``}Residential\text{''}}\textsc{Estate}, \textsc{Owner} \ltimes (\sigma_{Category=\text{``}Residential\text{''}}\textsc{Estate}), \textsc{Visit} \ltimes (\sigma_{Category=\text{``}Residential\text{''}}\textsc{Estate}), \textsc{Sale} \ltimes (\sigma_{Category=\text{``}Residential\text{''}}\textsc{Estate}), \textsc{Rent} \ltimes (\sigma_{Category=\text{``}Residential\text{''}}\textsc{Estate}), \textsc{Agenda} \ltimes (\sigma_{Category=\text{``}Residential\text{''}}\textsc{Estate}), \textsc{Picture} \ltimes (\sigma_{Category=\text{``}Residential\text{''}}\textsc{Estate})\}$$

Then for: $\texttt{C} = \langle \texttt{role} : \texttt{agent}(\$\texttt{ag\_id}), \texttt{category} : \texttt{residential} \rangle$

we can apply double intersection, and have:

$$\begin{aligned} \mathcal{R}el(\texttt{C}) &= \mathcal{R}el(\texttt{agent}(\$\texttt{ag\_id})) \Cap \mathcal{R}el(\texttt{residential}) \\ &= \{\sigma_{Category=\text{``}Residential\text{''}}\textsc{Estate}, \textsc{Owner} \ltimes (\sigma_{Category=\text{``}Residential\text{''}}\textsc{Estate}), \\ &\quad \textsc{Visit} \ltimes (\sigma_{Category=\text{``}Residential\text{''}}\textsc{Estate}), \textsc{Sales} \ltimes (\sigma_{Category=\text{``}Residential\text{''}}\textsc{Estate}), \\ &\quad \textsc{Rent} \ltimes (\sigma_{Category=\text{``}Residential\text{''}}\textsc{Estate}), \textsc{Picture} \ltimes (\sigma_{Category=\text{``}Residential\text{''}}\textsc{Estate}), \\ &\quad (\sigma_{IdAgent=\$ag\_id}(\textsc{Agenda})) \ltimes (\sigma_{Category=\text{``}Residential\text{''}}\textsc{Estate})\} \end{aligned}$$

More operator properties

Since both operators are *associative:*

- **If we use** *double intersection* **as the composition operator, then**

$$\mathcal{R}el(\langle V_1,\ldots,V_k,V_{k+1}\rangle) = \mathcal{R}el(\langle V_1,\ldots,V_k\rangle) \Cap \mathcal{R}el(V_{k+1})$$

- **If we use** *double union* **as the composition operator, then**

$$\mathcal{R}el(\langle V_1,\ldots,V_k,V_{k+1}\rangle) = \mathcal{R}el(\langle V_1,\ldots,V_k\rangle) \Cup \mathcal{R}el(V_{k+1})$$

i.e., we can obtain the view for a context *composed by k+1 context elements,* **by combining** *the partial view for the context formed by the first k elements* **with** *the view of the last one*

→ *Useful when the designer has to modify the CDT: if e.g. a dimension is added, relevant areas can be automatically re-computed*

→ *Favours dynamicity*

More operator properties

Since both operators are *commutative*, **we have:**

$$\mathcal{R}el(\langle V_1, V_2, \ldots, V_k \rangle) = \mathcal{R}el(\langle V_{i_1}, V_{i_2}, \ldots, V_{i_k} \rangle)$$

For any permutation of ***<1,…k>*** **, i.e., the view for a context composed by k context elements** *is independent of the order in which it has been composed*

More operator properties

Independently of the operator used for the composition, if:

- $C = \langle V_1, \ldots V_k \rangle$
- V, W **two context elements such that** $V \prec W$
- **we have:**

$$\mathcal{R}el(\langle V_1, \ldots, V, \ldots, V_k \rangle) \supseteq \mathcal{R}el(\langle V_1, \ldots, W, \ldots, V_k \rangle)$$

This is the extension to contexts of the assumption on containment for partial views of context elements

The NESTA case

The ART DECO project aims at supporting adaptive services and information in networked enterprises. The two case studies are the textile/fashion domain (NESTA), and the wine production domain (WINE). The NESTA database schema is given, and Context must be designed:

- Role
  - **Designer**
  - **Manufacturer (brand) (style executive, marketing executive)**
  - **Provider (fabric, accessories)**
  - **Distribution Agent**
  - **Retailer**
- Search Objective
  - **Product info (technical Info, semantic Info)**
  - **Commercial info**
- Project lifecycle phase
  - **Creative phase**
  - **Project Development**

bibliography

http://poseidon.elet.polimi.it/ca/