



Institute for Cartography and Geoinformation, University of Bonn
Chair for Geoinformation

Integrating Versions, History, and Levels-of-Detail within a 3D Geodatabase

Gerhard Gröger
Thomas H. Kolbe
Jörg Schmittwilken
Viktor Stroh
Lutz Plümer

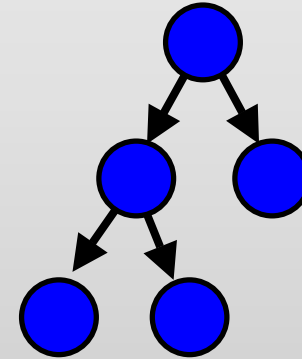
1st Int. Workshop on Next Generation 3D City Models, Bonn
June 21st, 2005

Overview

- Motivation
- Versions
- Workspace Concept of Databases
- City Model Aspects
- History
- Levels-of-Detail
- Conclusions

Versions, History, Level-of-Detail

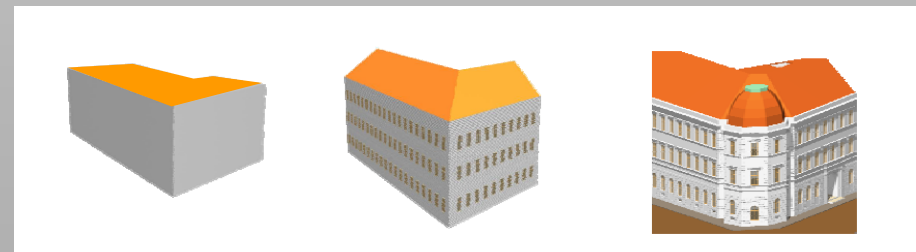
- Versions
 - spatial planning
 - alternative designs of same planning region
 - branching



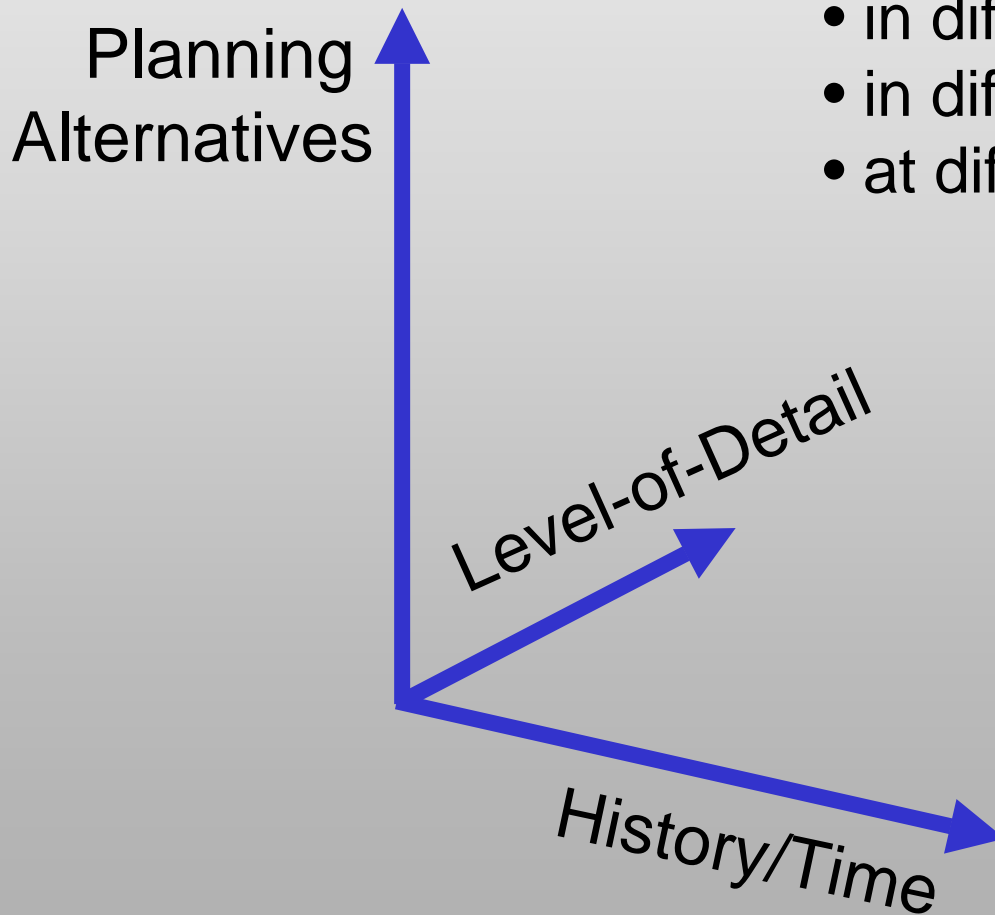
- History/Time
 - record all changes
 - e.g., state of the database on May 5th, 2003?
 - linear



- Levels-of-Detail
 - result of different collection processes



Dimensions of Multi-Representation



- The same object is represented
- in different planning alternatives
 - in different Levels-of-Detail
 - at different times

Integration: Example Query

- Select a specific part of Berlin,
 - with the **design A** in planning area X and the **design C** in planning area Y
 - at **May 21st, 2003**
 - in a **medium Level of Detail**

Questions

- How to **represent** the three dimensions of multi-representation **efficiently** and **conveniently** in a 3D geodatabase?
- How to **integrate** the three dimensions of multi-representation?
- Focus: Relational Database Oracle Spatial 10g

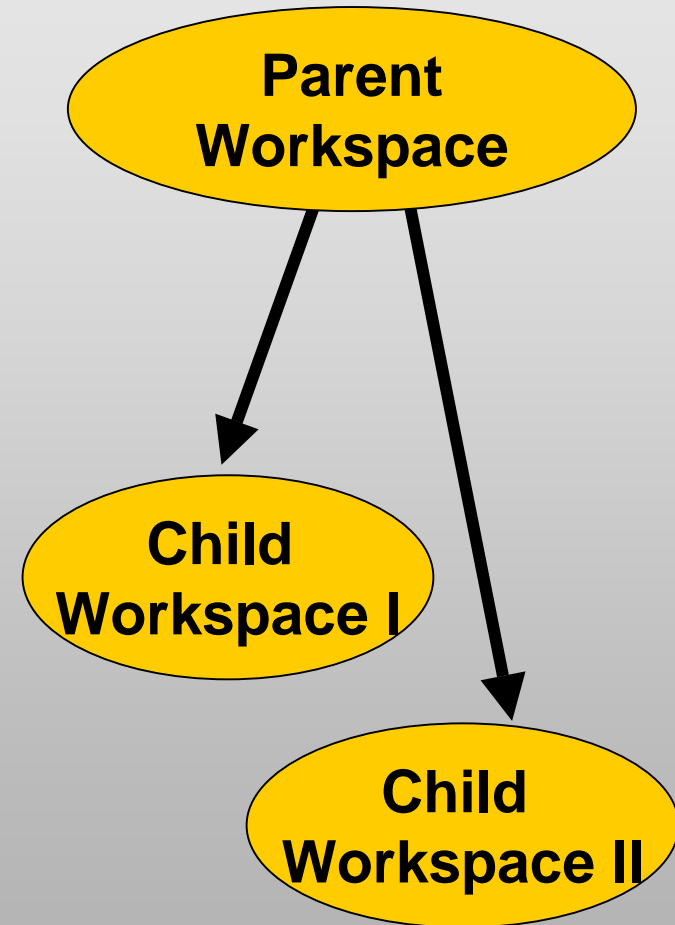


1. Implementation of Planning Versions

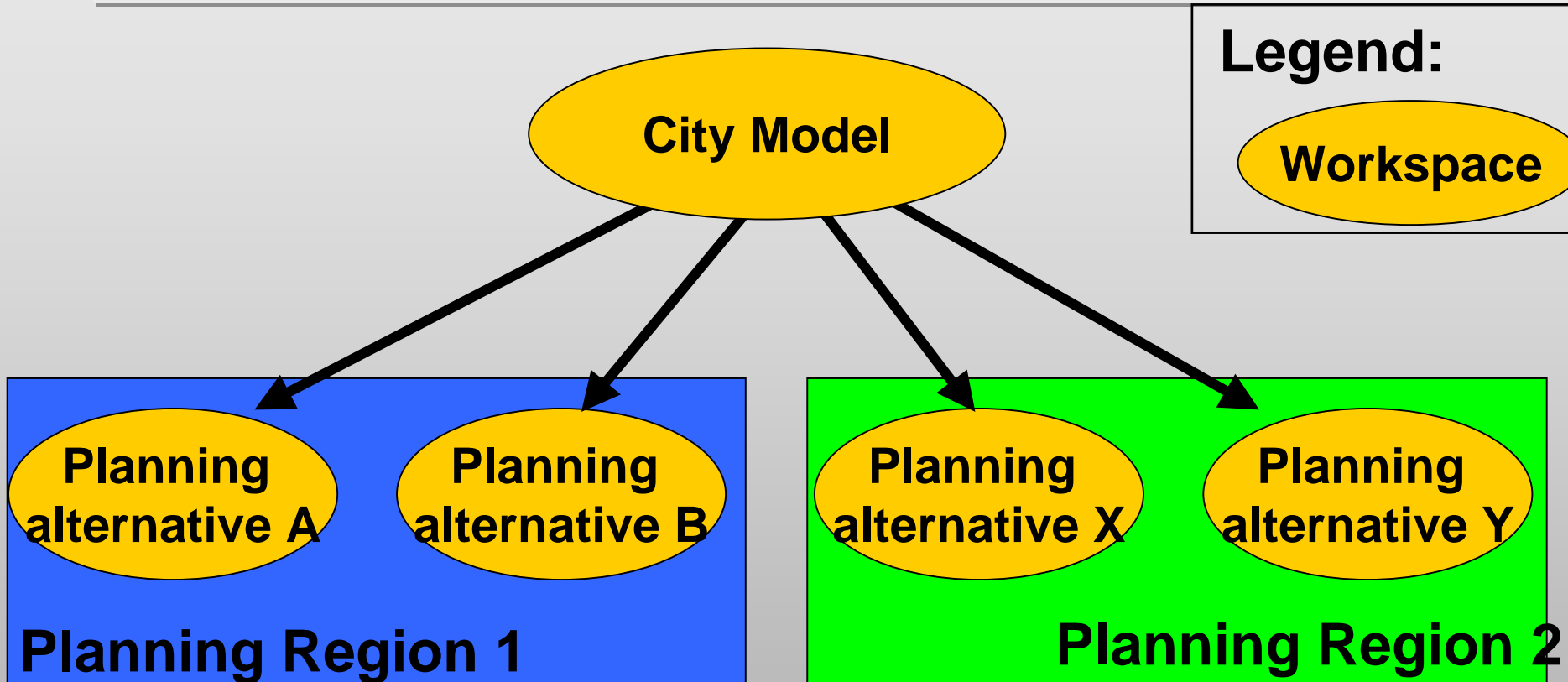
- Workspace concept

Workspace Concept

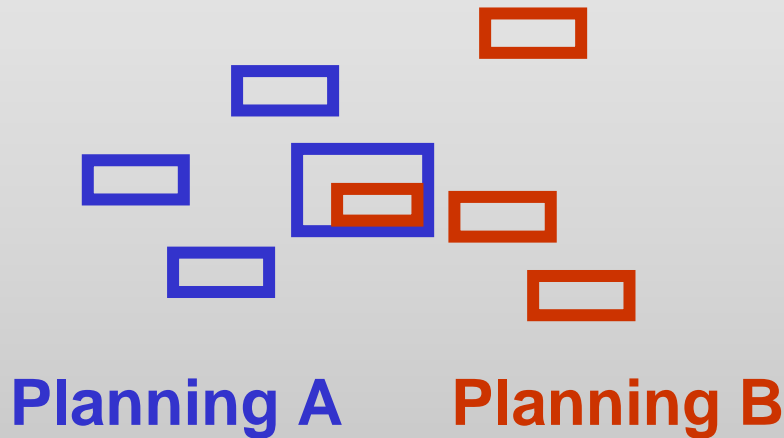
- database concept
- work on a "copy" of database, independently from others
- long-term transactions
- space efficient:
 - only changes/new objects stored explicitly
- transparent to user
- propagate modifications top-down or bottom-up
 - conflict handling



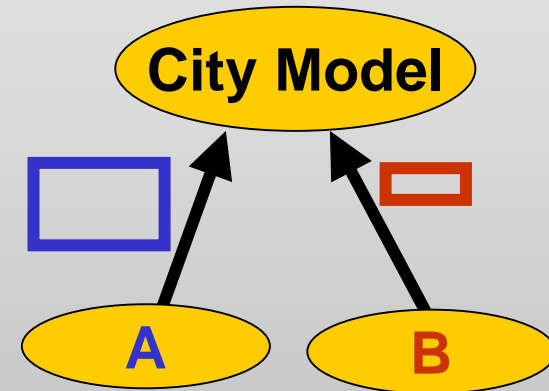
Spatial Planning using Workspaces



Planning Alternatives: Conflicts



Workspace Tree:



- Two Types of Conflicts:
 1. Modifications on the same Object (same ID)
⇒ **detected** by database
 2. Modifications of the same area (Insertion, Update)
⇒ **not detected** by database

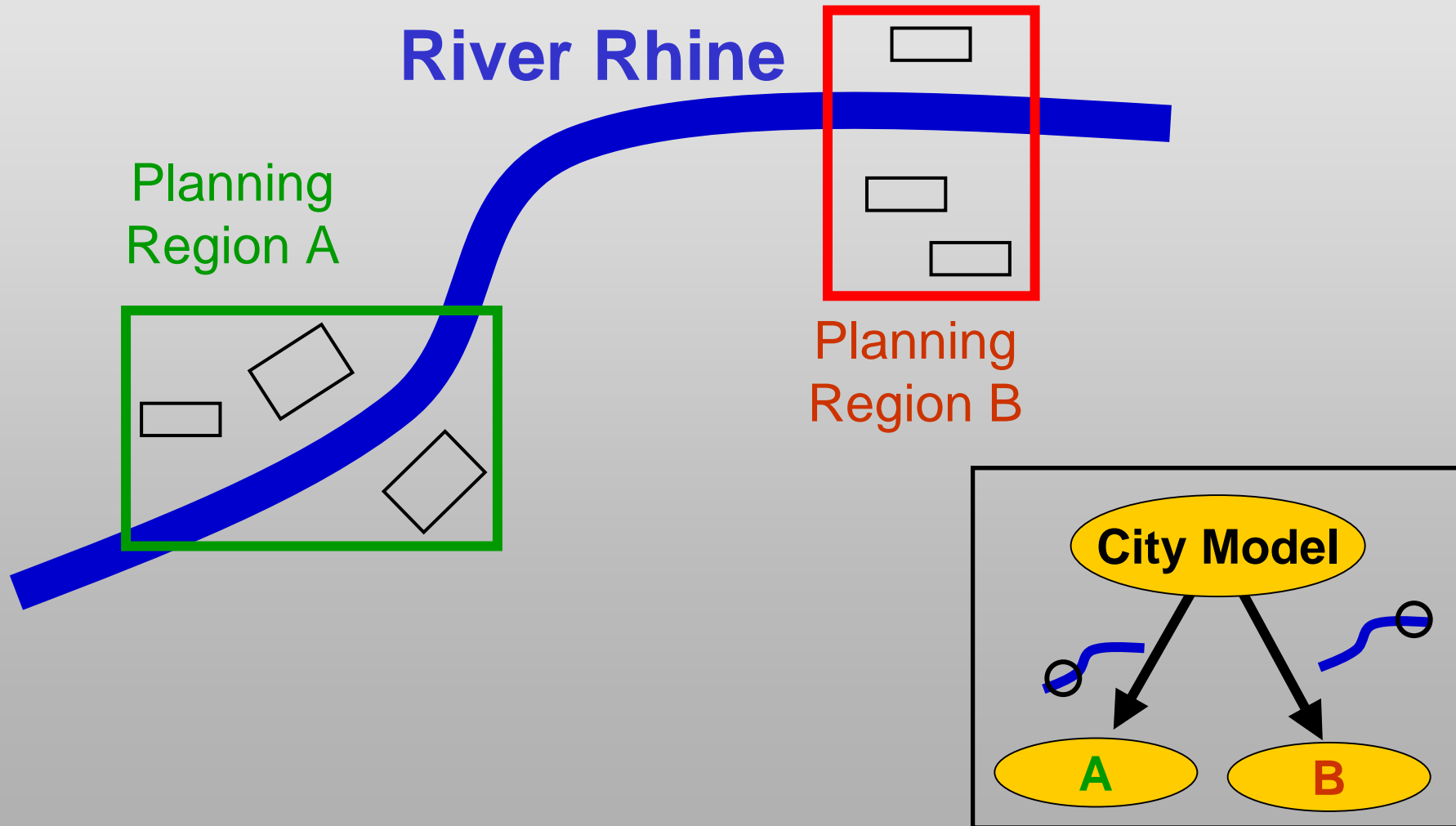


Avoiding Conflicts

- **Storing** of planning region **explicitly** (polygons)
- Planning Regions have to be **disjoint**
- **Check**: Is inserted/modified object **inside** corresponding planning region?
- **Database Triggers**
 - **Action** is fired automatically when **event** occurs and **condition** is met (**Event-Condition-Action**-Rule)
 - Example: When **geometry** of an object **is modified** and **geometry** is **outside planning region**, then **reject modification**
- Avoiding conflicts by **database mechanisms**

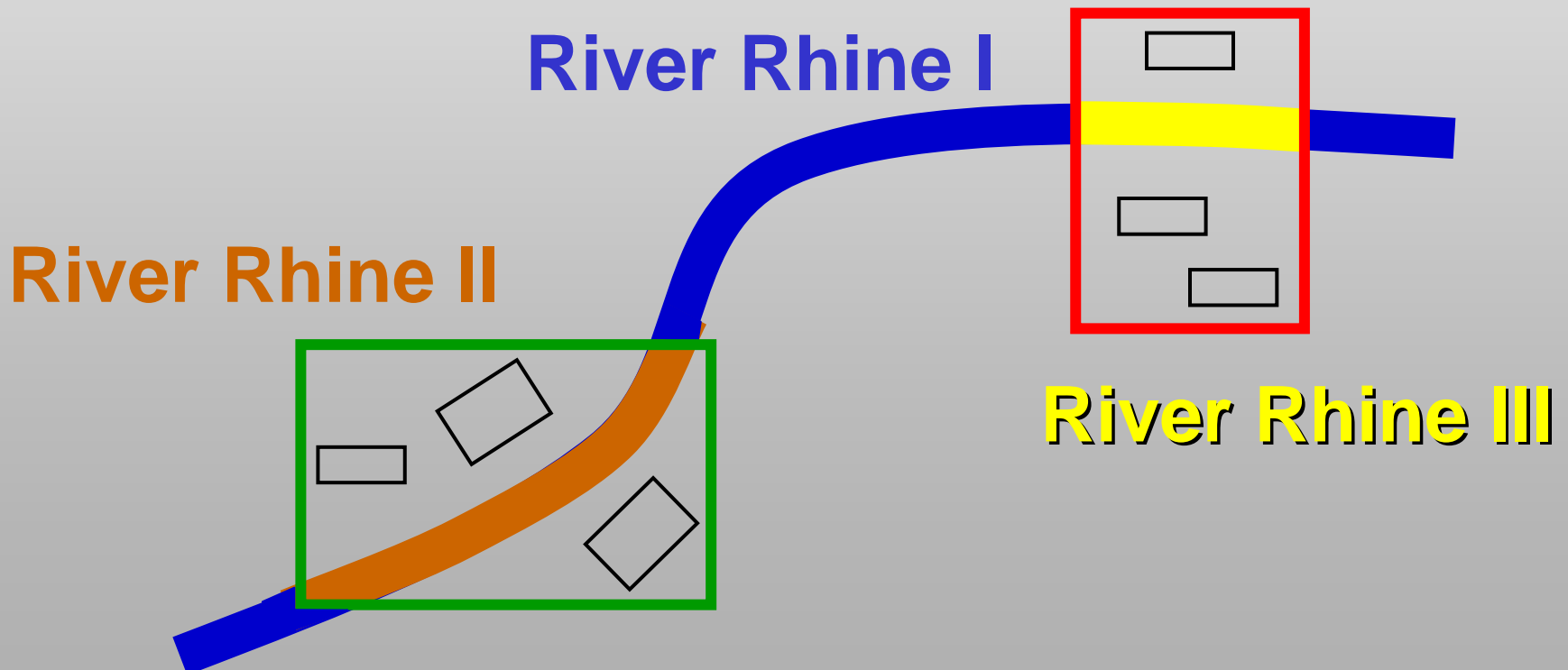


Conflicts with Large Objects: Example



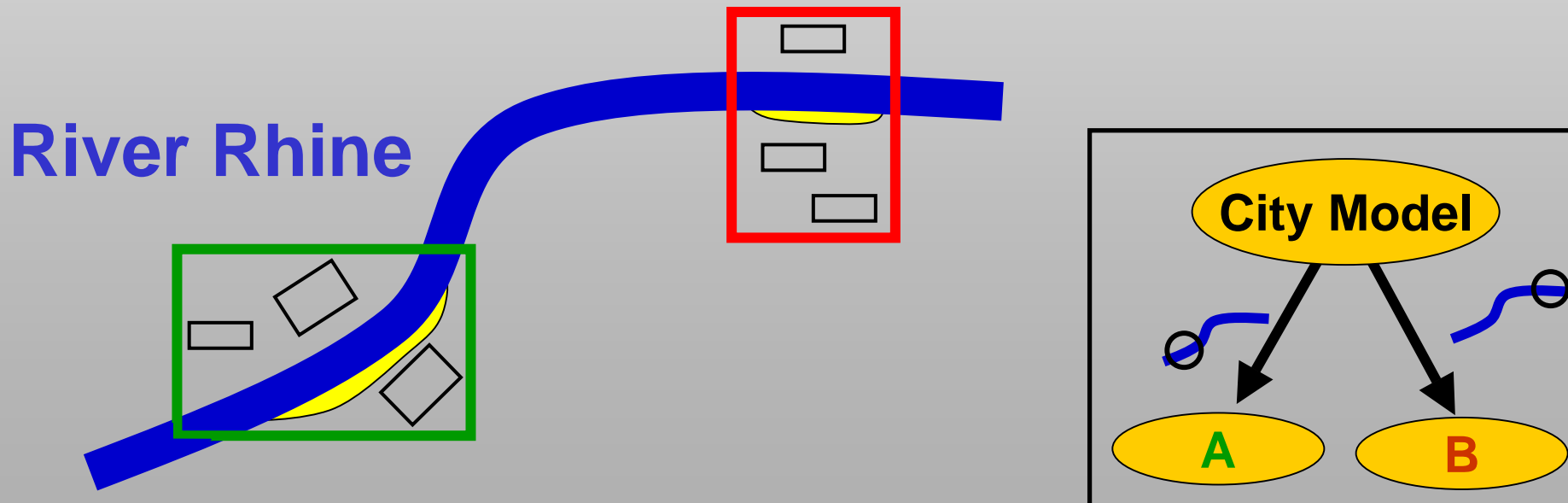
Conflicts with Large Objects

- Traditional solution: **Splitting** of large objects
 - Object homogeneity and identity lost
 - Updating of references



Conflicts with Large Objects

- More fine-grained approach
- **Check:** Is **inserted/modified geometry** inside corresponding planning region?
- Objects have to be **assembled** when merging



Different Versions: Integrated View

- different planning areas, each with different alternative designs
- users: customized views, combinations of different alternative designs
- **City Model Aspects**



City Model Aspect: Example



current
state

Planning
scenarios

A



B



X



Y



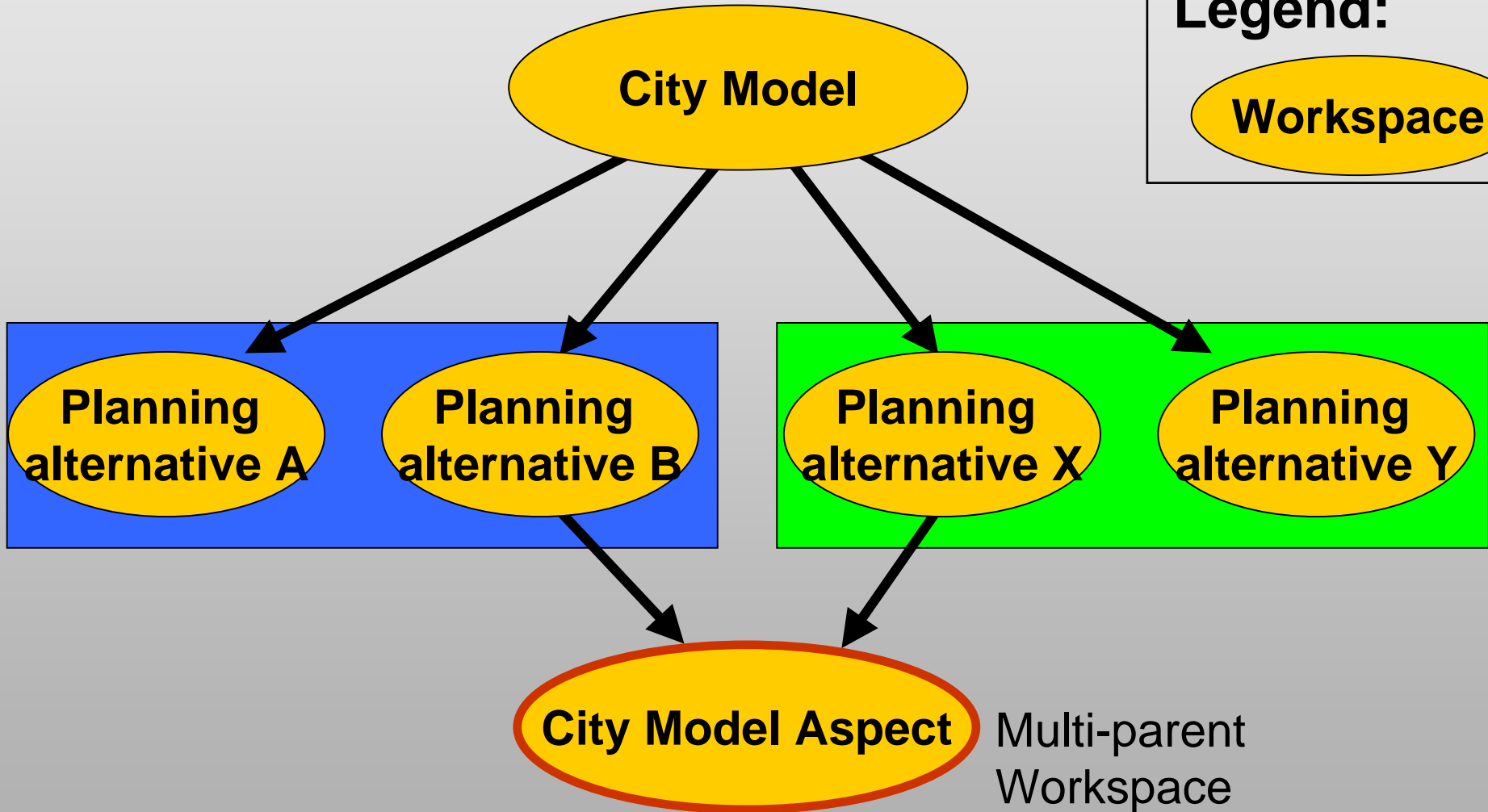
City Model
Aspect (B/X)



City Model Aspects: Implementation

Legend:

Workspace



2. Implementation of History/Time

- treated similar to versions
- using **workspace concept** of database
- offers method to set specific time or switch to specific time
- database operations **transparent** to user
- different semantics of time
 - database time (time of database modification)
 - "real" time (e.g., time of creation of "real" building)

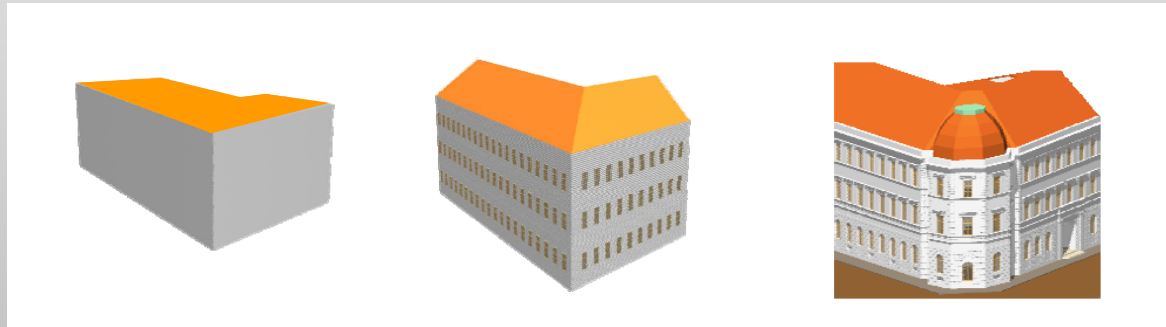


Database Oracle 10g: Deficiencies

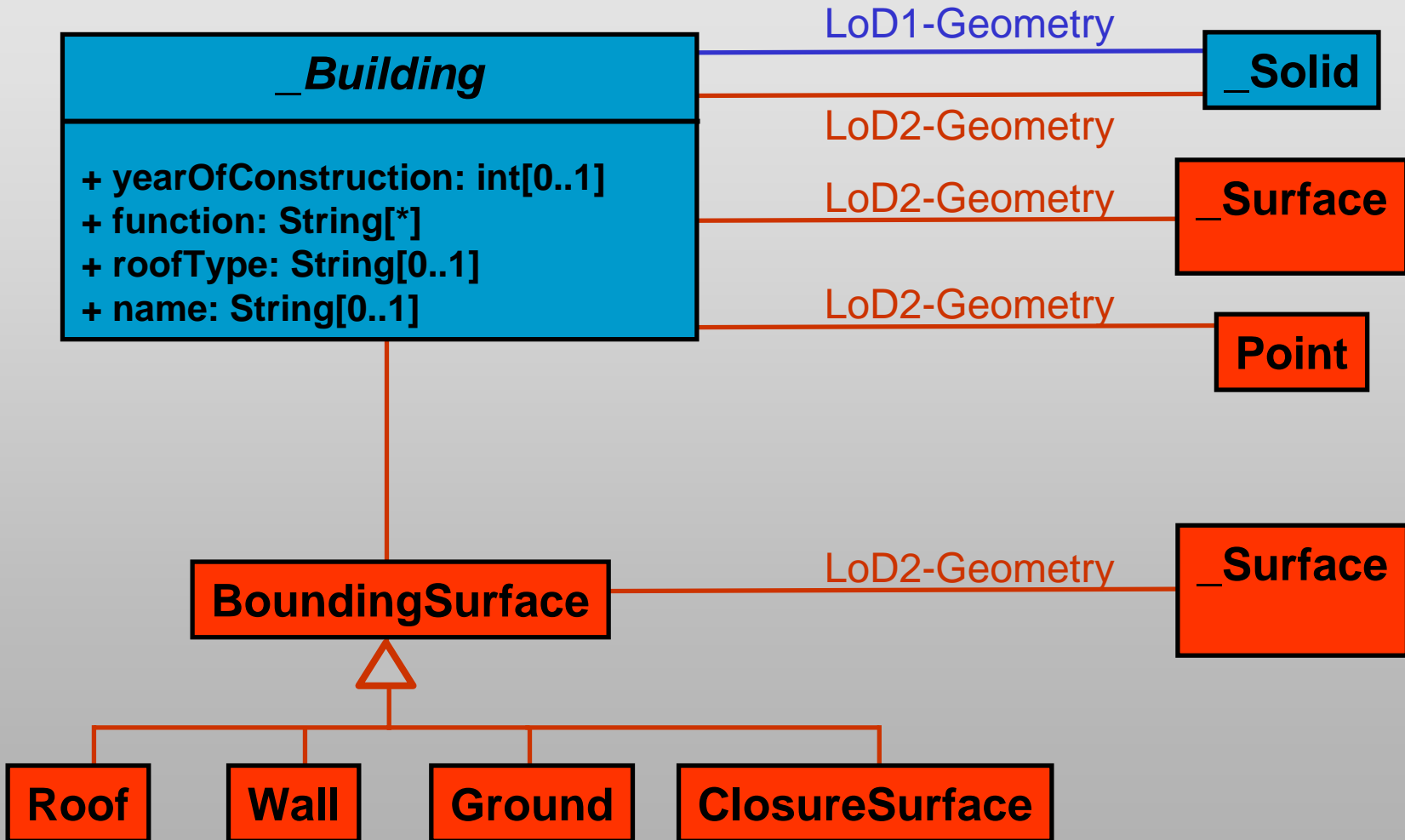
- **Workspace** concept **incompatible** with
 - **object-relational** concepts (object types, methods, inheritance, references, ..)
 - **Georaster** (georeferenced raster)
- **3D functionality** missing
 - consistency checks
 - avoiding of conflicts



3. Implementation of Levels-of-Detail



Building Model of CityGML (extract)



Level-of-Detail (LoD) vs. Versions/History

- Versions and History: **same model/schema** for each version/time
- LoD: **Model depends on level**, increasing LoD: more or more detailed classes
- deal with LoD on a **modeling level**, **not** on a **database level**



Context: Projects

- concept and implementation of a database for the 3D city model of Berlin
 - EU-funded project “Geodata management for the administration of Berlin – 3D VR model for investors and enterprises” in cooperation with the state administration and business development Berlin.
- CityGML: standard for 3D city models
 - SIG 3D (Special Interest Group 3D) of the Spatial Data Infrastructure North Rhine Westphalia (GDI NRW)
 - www.citygml.org



Conclusions

- **Orthogonal dimensions:** Planning Versions, History and Level-of-Detail
 - implemented in 3D spatial database
 - workspace concept for history/versions
 - **database level**
 - Conflict handling, fine-grained level
- **City Model Aspects:** Integrated Views
 - Level-of-Detail: treated on **modeling level**
 - Next steps: Evaluation on large 3D data sets, add 3D functionality to database

Thank you for your attention.

Any questions?