

CapView – Functionality-Aware Visual Mashup Development for Non-programmers

Carsten Radeck, Gregor Blichmann, Klaus Meißner

Talk outline

- 01 Problem Statement
- 02 Conceptual Foundation
- 03 CapView
- 04 Evaluation
- 05 Conclusion & Future Work



<http://mmt.inf.tu-dresden.de/EDYRA>

End user development (EUD) gains momentum to fulfill long tail user needs

Mashups as potential solution, but

- still very technical metaphors, terms
- distinction development time ↔ run time
- ...

Non-programmers as target group:

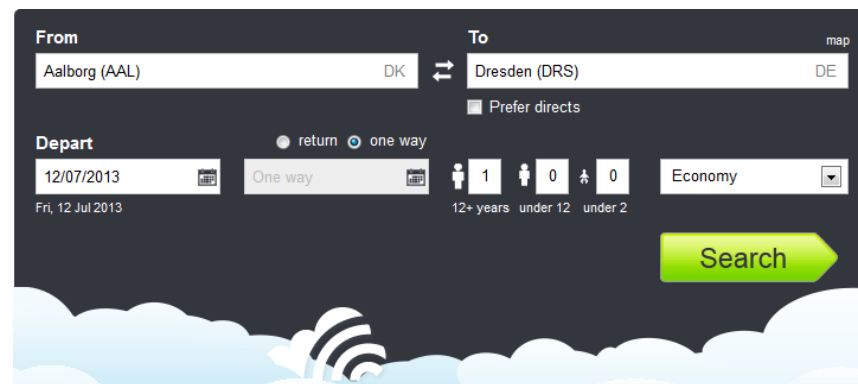
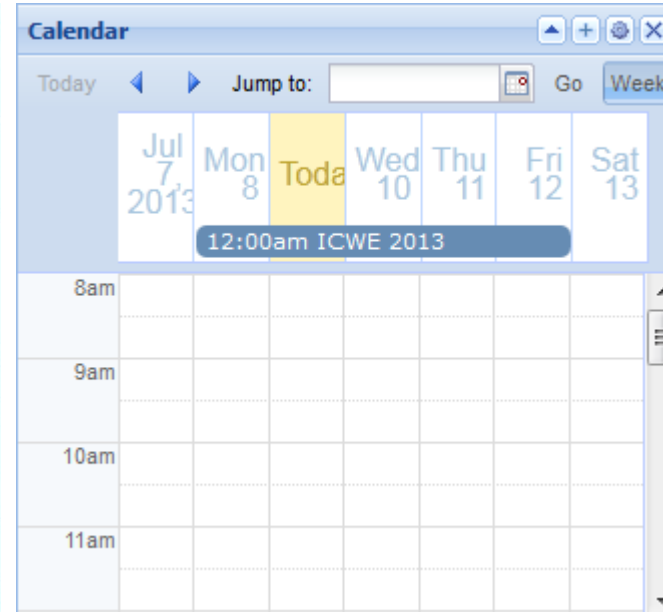
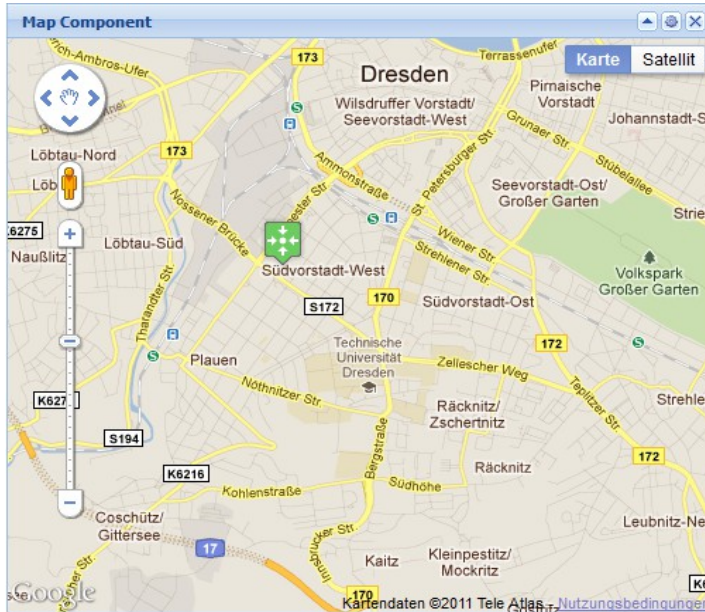
- limited understanding of technical concepts
- no experience on development practices



[P1]

→ It is hard for the user to map his/her problem to a composition
→ How to understand what happens in a mashup?

01 Problem Statement



General challenges for EUD tools [6]

- abstraction of technical details & terminology
- automation, short feedback loops
- user guidance

Challenge we address

- appropriate level of abstraction
 - to clarify functionality mashup (parts) and recommendations provide
 - for composing on a task/activity-level

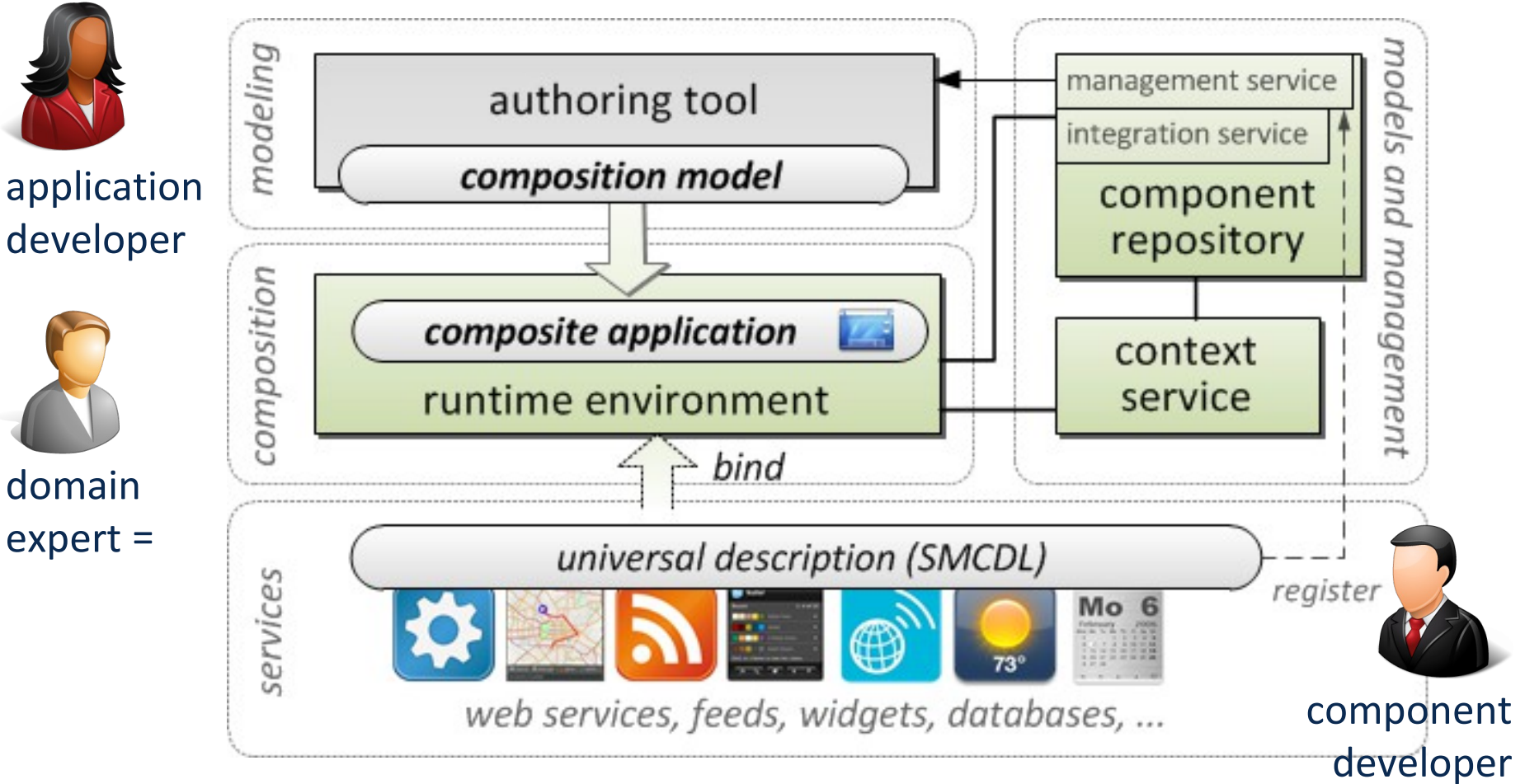


[P2]

→ *CapView*

- helps to realize “components” as task-solving entities,
- investigate functionalities provided by a mashup,
- manipulate a mashup by visually composing functionalities

End User Development for Mashups: EDYRA [4]



Based on  CRUISe [2]

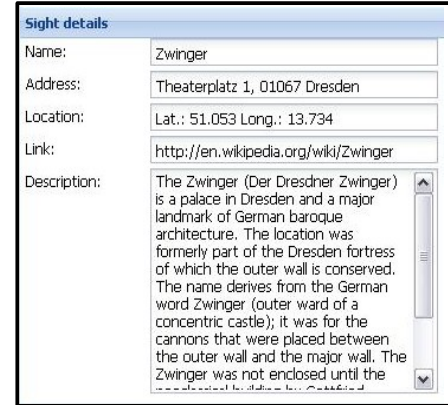
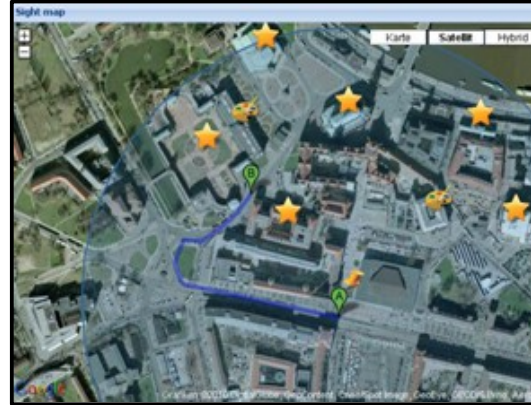
Component Model

- uniform black-box view
- parametrized **operations** and **events**, **properties**
- semantic component description (SMCDL [3])

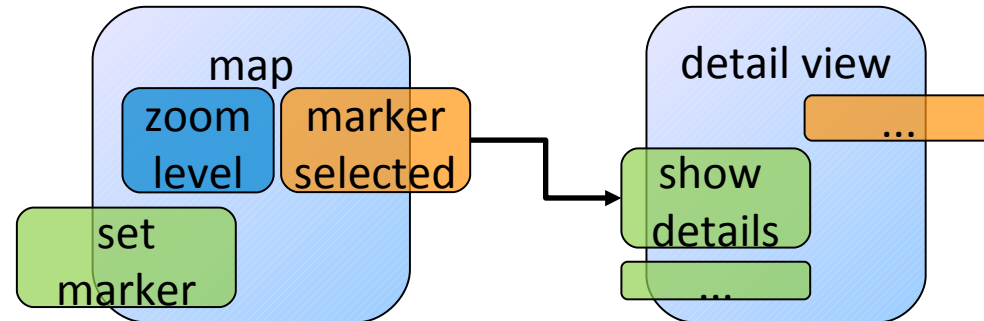
Composition Model

- covers application aspects
- included components,
- communication,
- layout, navigation,...

final mashup



composition model



web APIs



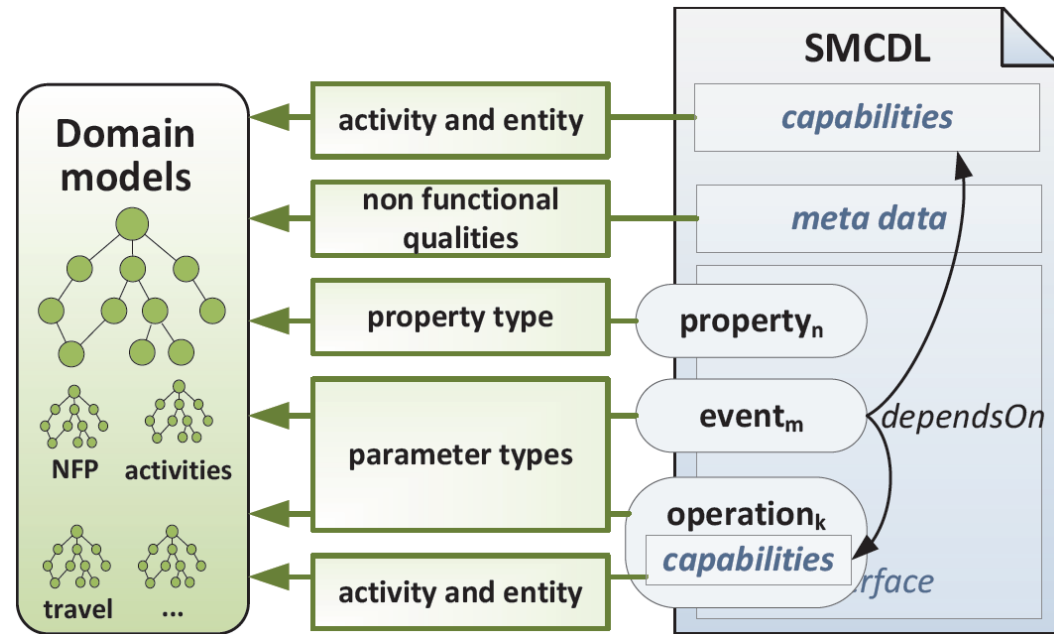
Google Maps



Google Places

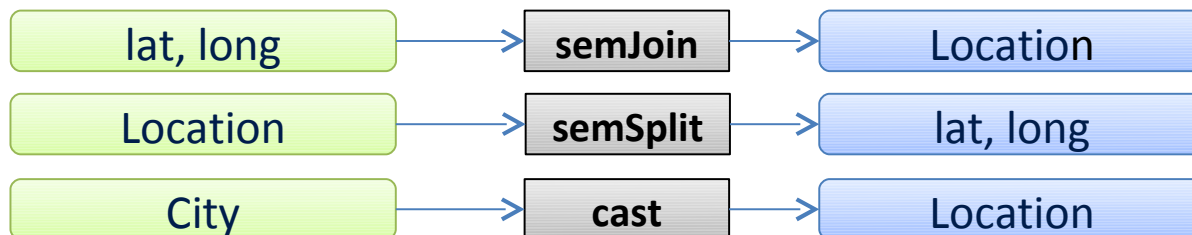
Capabilities

- for functional semantics of interface elements
- (*activity, entity, requiresInteraction*)
- e. g. **Display Location**



Mediation Techniques extending [3]

- resolve data heterogeneity
- reflect knowledge of referenced ontologies, e.g.:

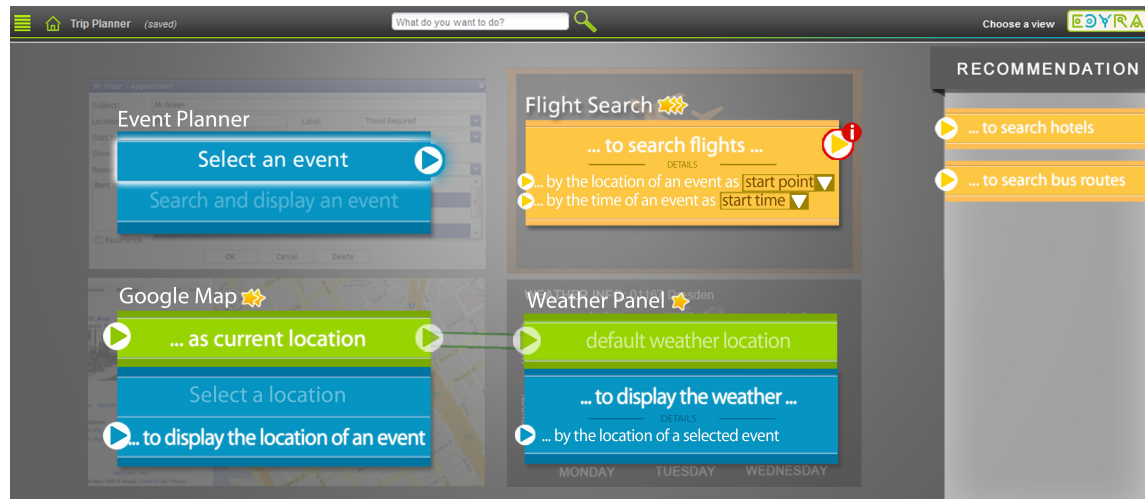
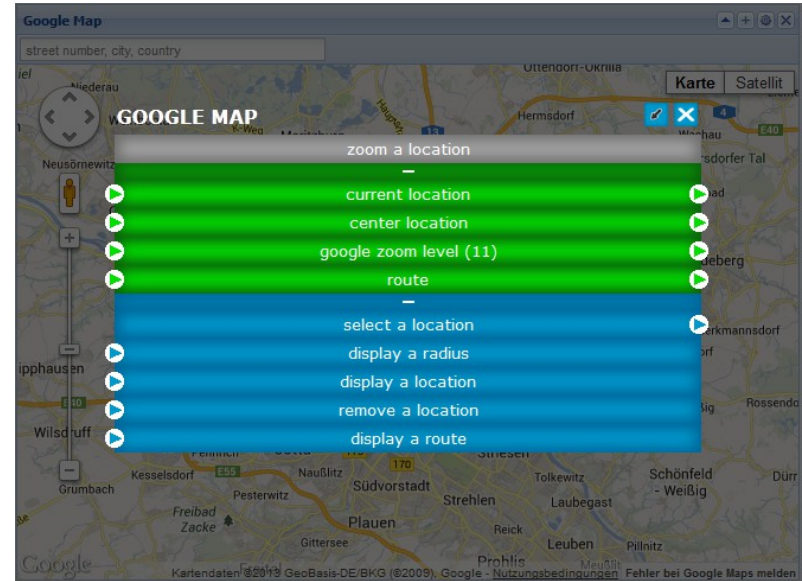


Assumption

- components serve to solve tasks
 - need input, provide output

A view that overlays a running application

- representations of capabilities and properties are visualized
 - labels
 - ports for connecting
- spatial correlation
- recommendations

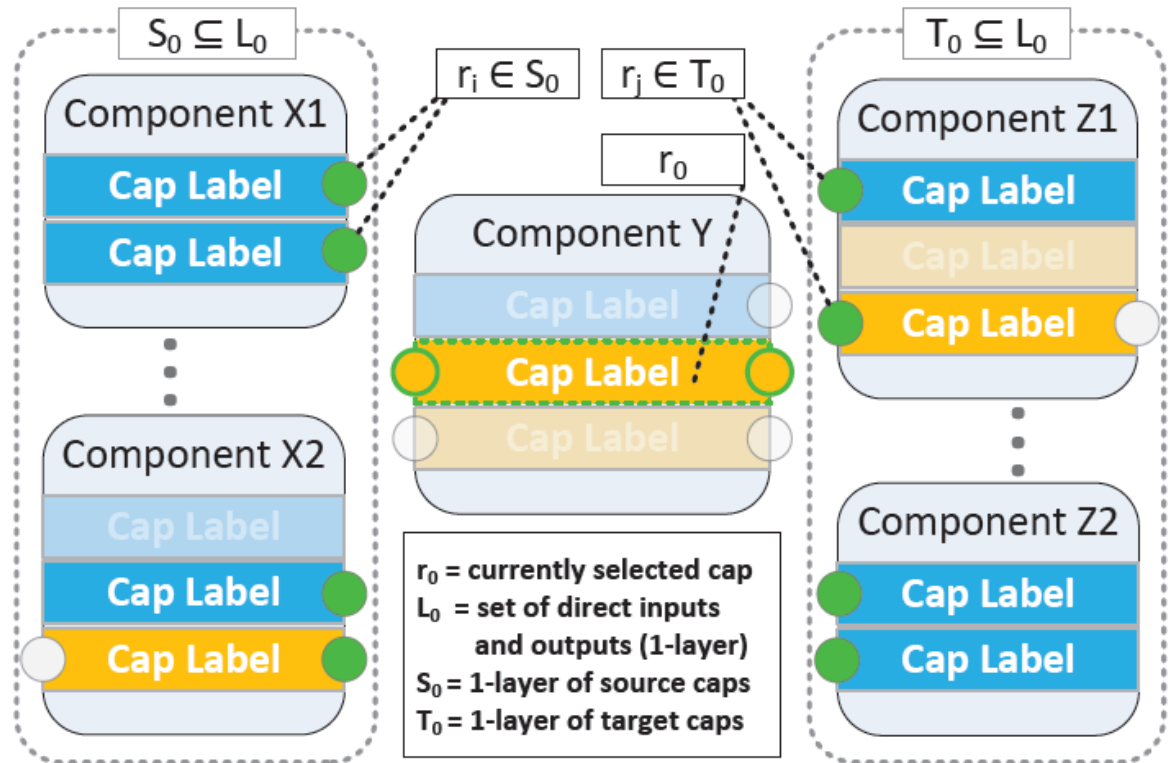


Color coding

- properties: uniform
- capabilities: depending on *requiresIteration*

Formal definitions

- r_0 - active selection
- L_0 - all r connectable with r_0
- $S_0 \subseteq L_0$ and $T_0 \subseteq L_0$

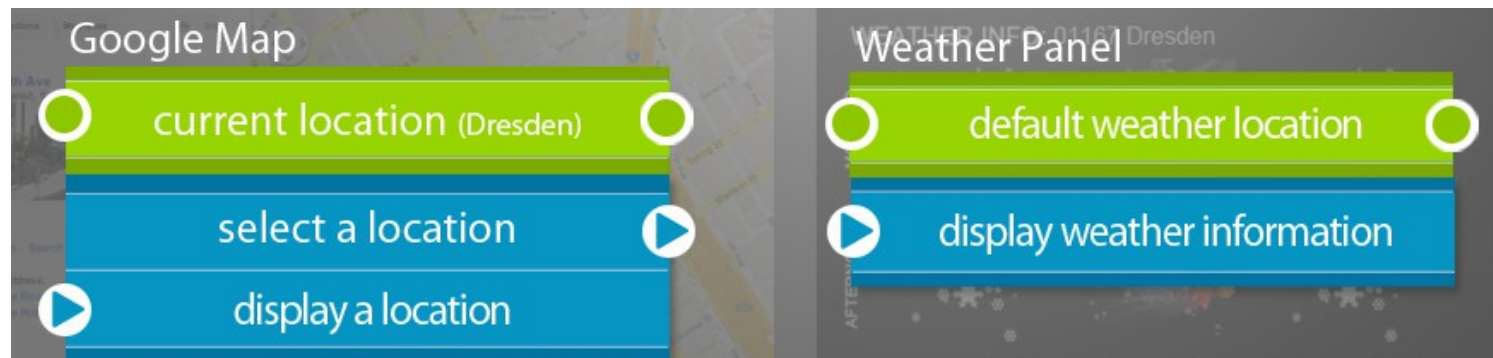


Upon user selection

- highlighting and re-labeling all r in L_0
- direct and transitive connections are highlighted

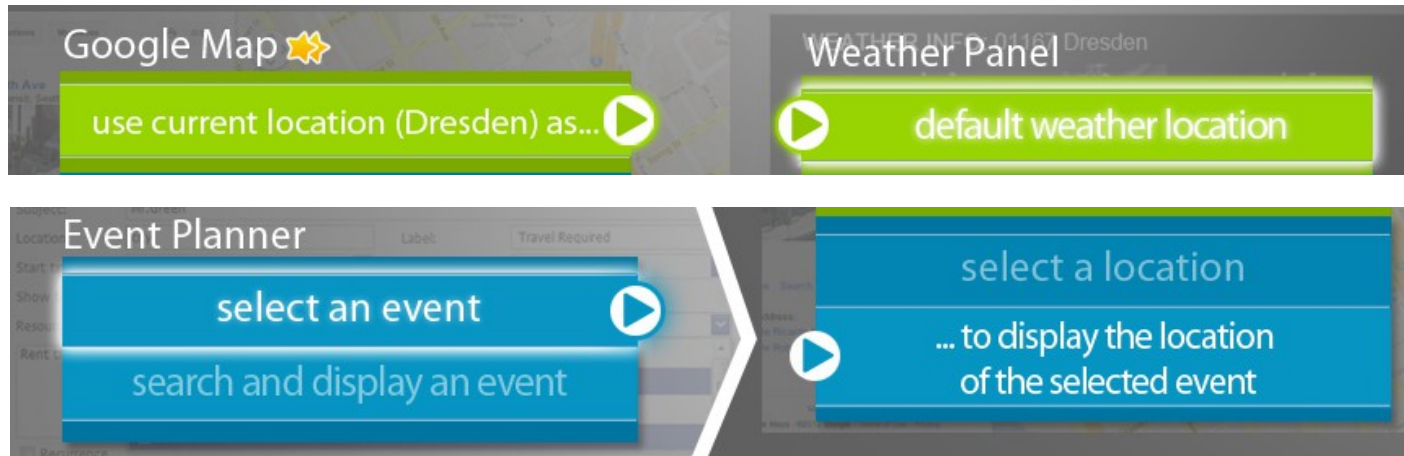
Generic rule set for deriving labels based on annotations

- used by label generator
- in case nothing is selected:
 - property: concept name extracted from the URI or `rdfs:label` and instance data if set
 - capability: short phrase combining activity and entity, concept name extracted from the URI or `rdfs:label`



Details on rules when active user selection

- label of r_0 not changed
- clarify cause and effect → build phrases
 - prepend or append dots
 - treat r_j in T_0 and r_i in S_0 differently
- rules differ slightly depending on whether involved r represent properties or capabilities



The image shows four examples of context-sensitive labels overlaid on different application screens:

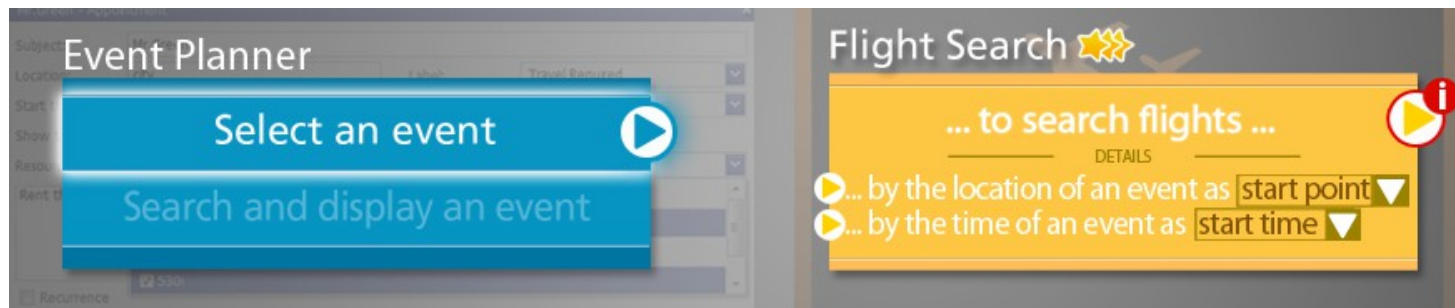
- Google Map:** A green label reads "use current location (Dresden) as...".
- Weather Panel:** A green label reads "default weather location".
- Event Planner:** Two blue labels are shown: "select an event" and "search and display an event".
- Event Planner (continued):** A blue label reads "select a location ... to display the location of the selected event".

Details on rules when active user selection

- special rules take semantic relation of entities and parameters into account
 - **SplitRule**
 - target parameter type is subsumed by source parameter type
select event *display location*
select event → ... to display **the location of the selected event**
 - **CompRule**
 - if entity and parameter of capability are mediable → shorten
 - **SuffixRule**
 - If there are multiple possible parameter mappings → options
center *search route* (startLocation, destLocation)
center → *search a route using the center* **as start/as dest.**
- combinations possible

Establishing connections

- requires active selection
- in case of multiple parameters and/or multiple possible mappings: choose options



- connections can be created from source to target and vice versa
- via drag & drop or clicking the target port
- runtime environment implements all composition model details

Setup

- preliminary prototype
- 10 students from fields in the age of 22 – 37
- questionnaire (demographic data and skills)
 - ✓ no or very basic knowledge about mashups
 - ✓ frequently use web applications
 - ✗ 5 users described their programming skills as “average”
- introduction by interviewer
- think aloud protocol
- after completion: assess perceived task load and system usability

Fragebogen

Demographische Angaben

Alter: _____

Geschlecht: _____

Beruf: _____

1 2 3 4 5

Wie schätzen Sie Ihren Erfahrungslevel mit Computern ein?
(1 = keine Erfahrung, 3=mittel, 5=sehr viel Erfahrung)

Haben Sie Erfahrung mit ~~Mashup-Technologien~~? _____

kein grundlegend fortgeschritte professionel

Welche Programmierkenntnisse besitzen Sie?

Was ist ihr Fachgebiet (Studiengang, Branche o.ä.)? _____

Bewertung des Systems

Markieren Sie für jede Aussage eine Box, die Ihre Reaktion bezüglich des Systems am besten beschreibt. Treffen Sie die Wahl möglichst spontan. Bei Unsicherheit ist das mittlere Feld anzukreuzen.

Stimme klar Stimme klar
nicht zu zu

1.) Ich denke, ich würde dieses System gerne häufiger benutzen.

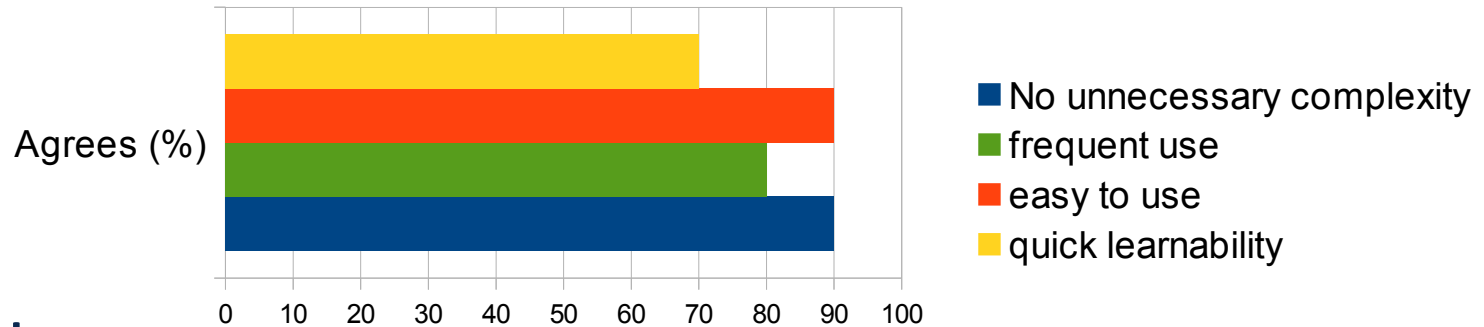
Positive points

- ✓ all participants were able to solve the tasks
- ✓ key concepts perceived positive
 - functional abstraction
 - spatial correlation
 - highlighting
 - natural language labels & context-sensitive relabeling
- ✓ labels sufficiently intuitive (70%)
- ✓ both directions used for connection creation

Task Load Index:



System Usability Scale [5]: 78,5 (70 – 92,5)



Neutral assessment

- ✘ port metaphor
- ✘ color coding minor role

Problems

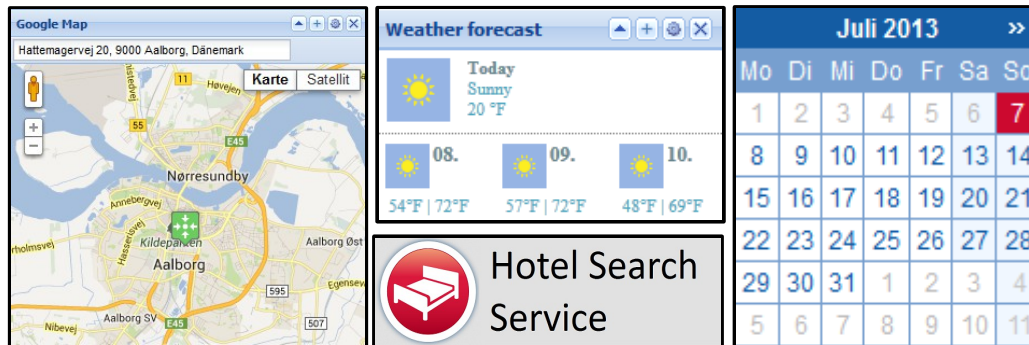
- ✘ concept of service components hard to understand
- ✘ abstraction from instance data
- ✘ expectations on component functionality highly influenced by experience

Proposal: CapView

- View on a mashup under development/usage
 - emphasizing capabilities
 - abstracting composition details
 - generating and adapting natural language labels
 - forming sentences to explain functional interplay
 - seamlessly integrating recommendations
- evaluation by means of prototypical implementation, and user study

Current and Future Research Focus

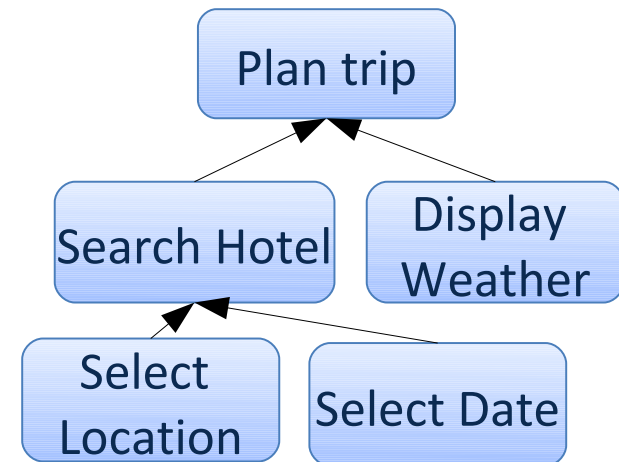
- stronger interweaving CapView and LiveView
- intuitive way for users to express their goal
- derive capabilities and their relationships from complex composition fragment

The screenshot displays three main components: a Google Map of Aalborg, Denmark; a weather forecast for the next three days (08. to 10. July); and a 'Hotel Search Service' button with a red icon of a bed.

Juli 2013						
Mo	Di	Mi	Do	Fr	Sa	So
1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28
29	30	31	1	2	3	4
5	6	7	8	9	10	11

- conduct a comparative user study



Thank you for your attention!

Questions?

Carsten Radeck
carsten.radeck@tu-dresden.de
Skype: c.radeck

Funding for the EDYRA project is provided by the Free State of Saxony and the European Union within The European Social Funds program (ESF-080951805).



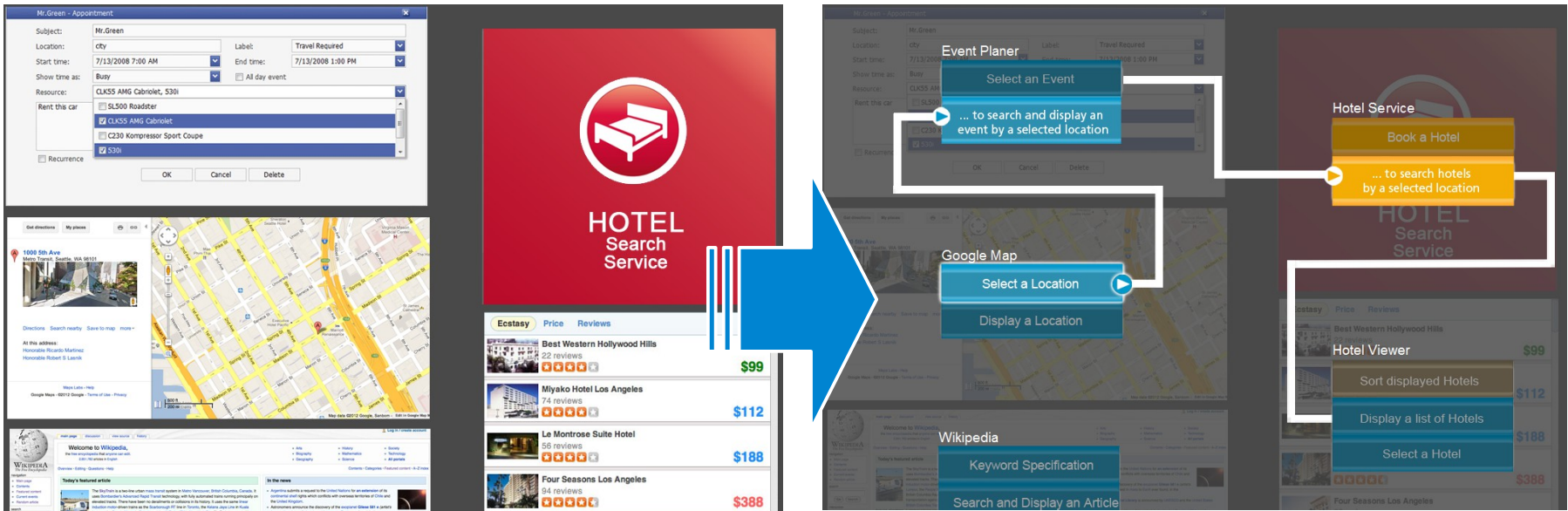
Tasks to be solved (in travel planning domain)

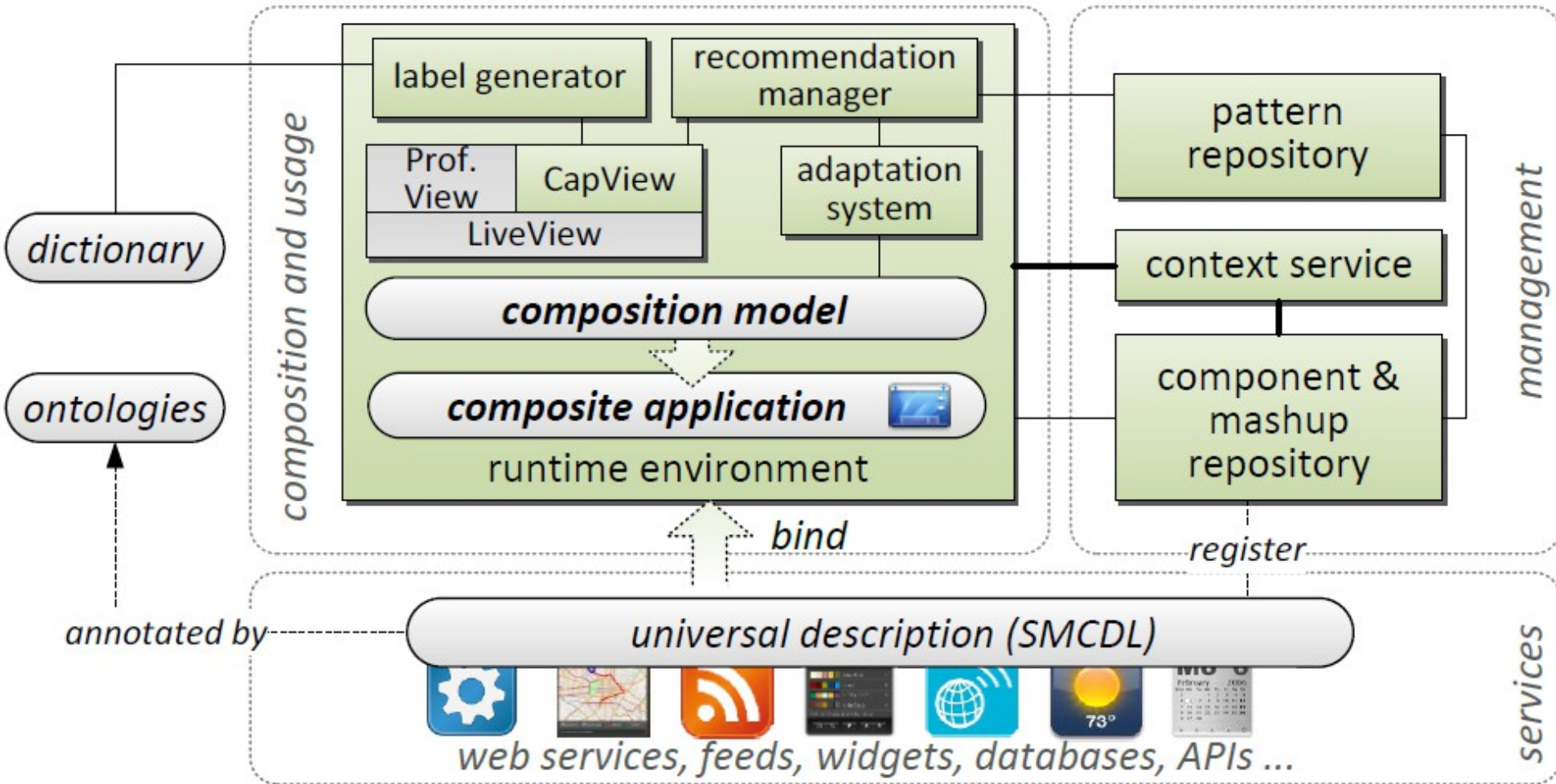
Scenario 1

- basic understanding of the exploration and interaction mechanisms

Scenario 2

- creating and reconfiguring connections using parameter mappings





- [1] Carsten Radeck, Alexander Lorz, Gregor Blichmann, and Klaus Meißner: **Hybrid Recommendation of Composition Knowledge for End User Development of Mashups** In *Proc. of the 7th Intl. Conf. on Internet and Web Applications and Services (ICIW 2012)*, 2012.
- [2] Stefan Pietschmann: **A Model-Driven Development Process and Runtime Platform for Adaptive Composite Web Applications**. In *International Journal On Advances in Internet Technology (IntTech)*, vol. 4, no. 2, 2009 (published March 2010).
- [3] Stefan Pietschmann, Carsten Radeck, and Klaus Meißner: **Semantics-Based Discovery, Selection and Mediation for Presentation-Oriented Mashups**, In *Proceedings of the 5th International Workshop on Web APIs and Service Mashups (Mashups 2011)*, ACM, September 2011
- [4] Andreas Rümpel, Carsten Radeck, Gregor Blichmann, Alexander Lorz, Klaus Meißner: **Towards Do-It-Yourself Development of Composite Web Applications**, In *Proc. of the International Conference on Internet Technologies & Society 2011*, 2011
- [5] Brooke, J.: **SUS: a “quick and dirty” usability scale**, In *Usability Evaluation in Industry*, 1986
- [6] Abdallah Namoun, Tobias Nestler, and Antonella De Angeli: **Service Composition for Non-programmers: Prospects, Problems, and Design Recommendations**, In *Proceedings of the 8th European Conference on Web Services, Cyprus*, pp. 123–130, 2010
- [P1] <http://www.photocase.de/stock-fotos/226003-stock-photo-mensch-mann-pflanze-spielen-garten-kopf.jpg>
- [P2] <http://www.photocase.de/foto/213685-stock-photo-mann-computer-hand-jugendliche-arbeit-erwerbstaetigkeit-gefuehle>