Supporting Component Upgrades with Metadata

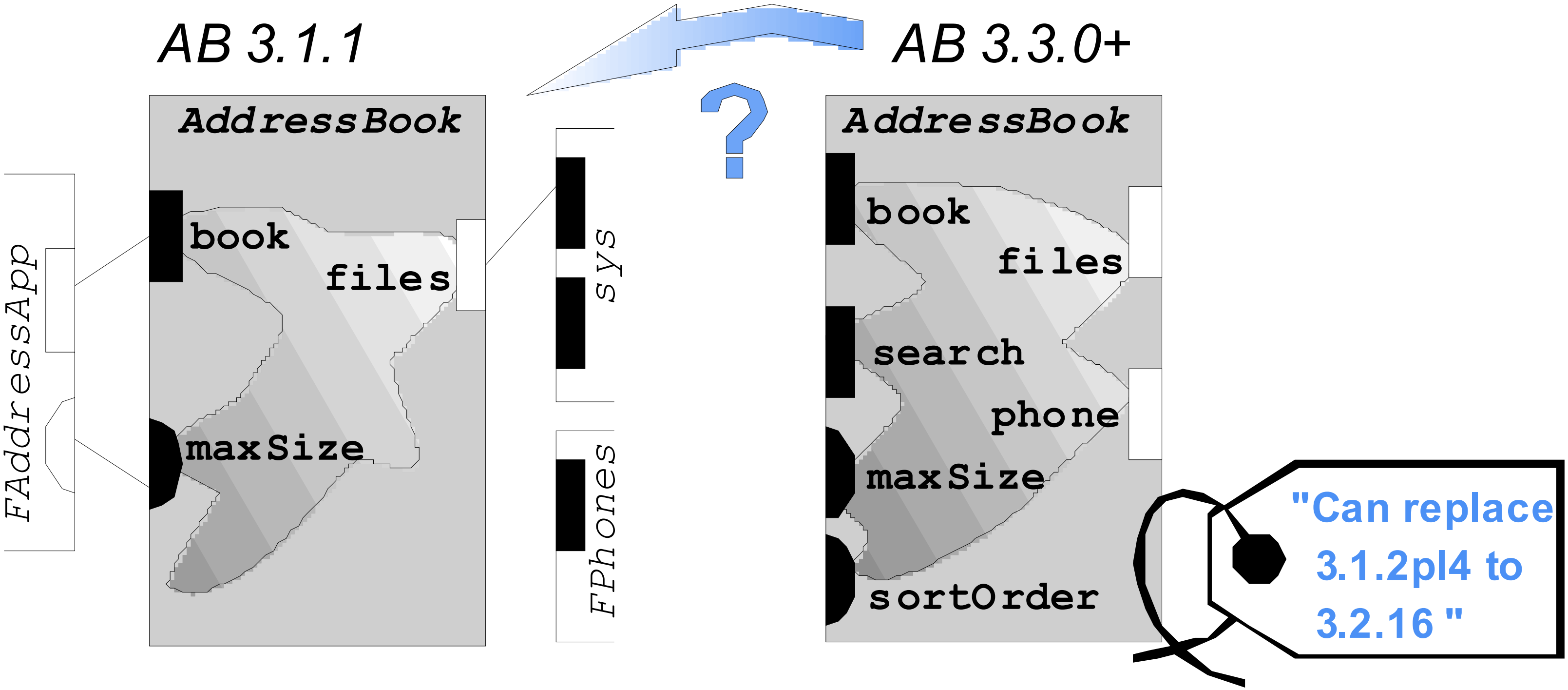
Přemysl Brada
University of West Bohemia
Pilsen, Czech Republic
<brada@kiv.zcu.cz>

Supporting Component Upgrades with Metadata

Overview of the Talk

- The Problem
- Specification → Model
- Differences → Revisions → Metadata
- Pre-update checking process

Problem: Component Upgrade

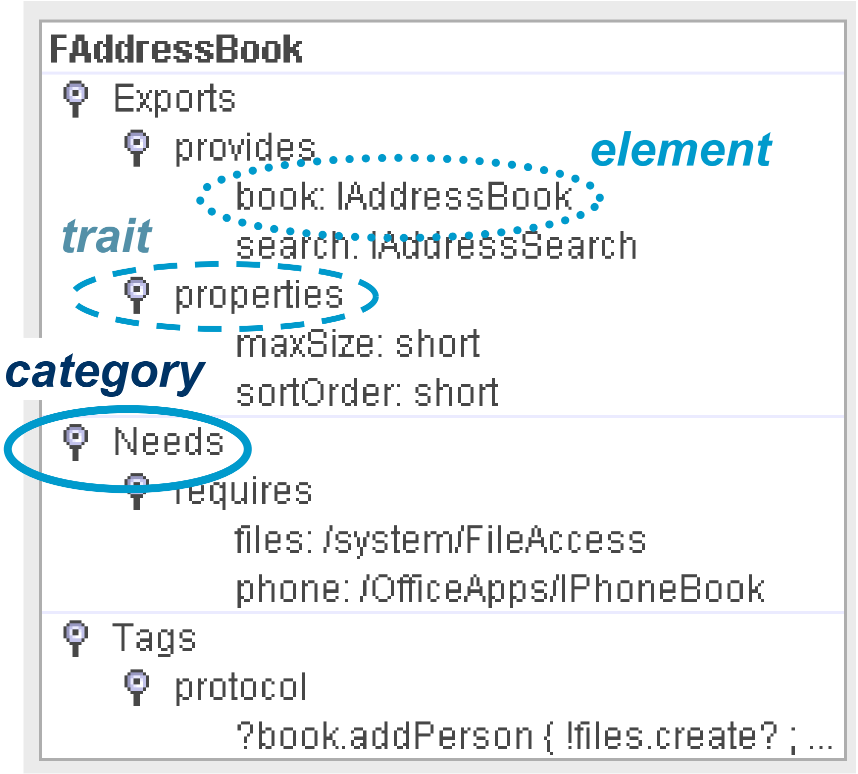


- Manually obtained compatibility data
  - error prone, added work

ENT Component Metamodel

specification (binary) → model (analyses, visualization)

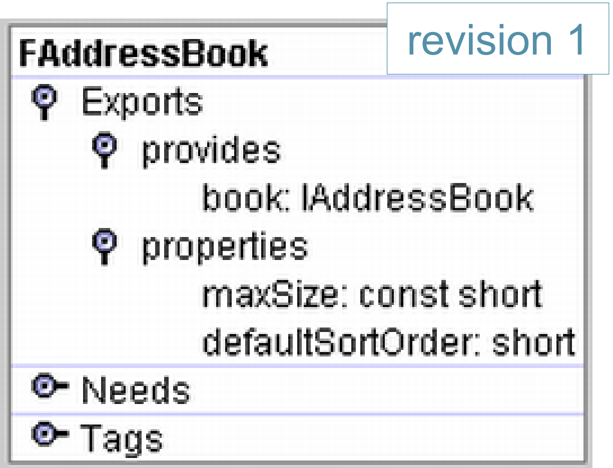
```
frame FAddressBook {
 provides:
   IAddressBook book;
   IAddressSearch search;
 property short sortOrder;
 requires:
   /system/FileAccess files;
   /OfficeApps/IPhoneBook
     phone;
 property short maxSize;
 protocol:
   ?book.addPerson {
   !files.create? ;
   !files.write }
   ...
}
```



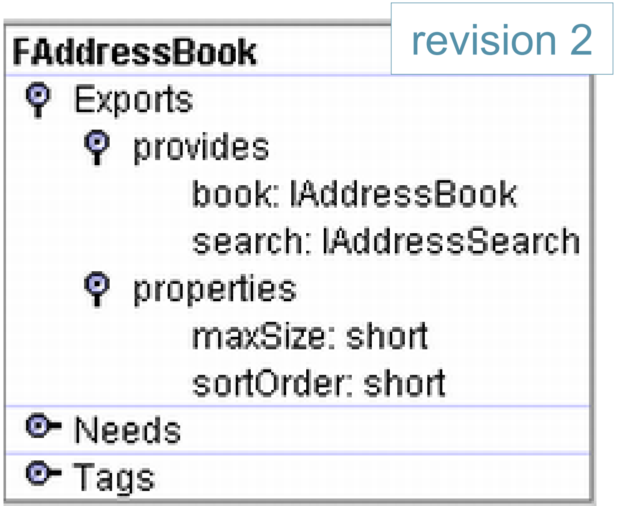
FAddressBook
- Exports
  - provides
    - book: IAddressBook
    - search: IAddressSearch
  - properties
    - maxSize: short
    - sortOrder: short
- Needs
  - requires
    - files: /system/FileAccess
    - phone: /OfficeApps/IPhoneBook
- Tags
  - protocol
    - ?book.addPerson { !files.create? ; ...

property classification:
content=*feature*, role=*provided*, kind=*data*, lifecycle=*assembly*

ENT Diff: Changes Classified


revision 1

FAddressBook
- Exports
  - provides
    - book: IAddressBook
  - properties
    - maxSize: const short
    - defaultSortOrder: short
- Needs
- Tags


revision 2

FAddressBook
- Exports
  - provides
    - book: IAddressBook
    - search: IAddressSearch
  - properties
    - maxSize: short
    - sortOrder: short
- Needs
- Tags

- ENT structures: what is different
  - changes in elements, traits, categories
- Classification: how did it change
  - *specialization* = subtype “$e_2 < e_1$”
  - *generalization* = supertype “$e_2 > e_1$”
  - *mutation* = incomparable “$e_2 \Delta e_1$”

Revision + Diff = Metadata

- **Revision IDs:**
  - indicate **where** a change occurred in ENT structures (3.1.1 → 4.2.1)
  - each `revnum` incremented on any difference

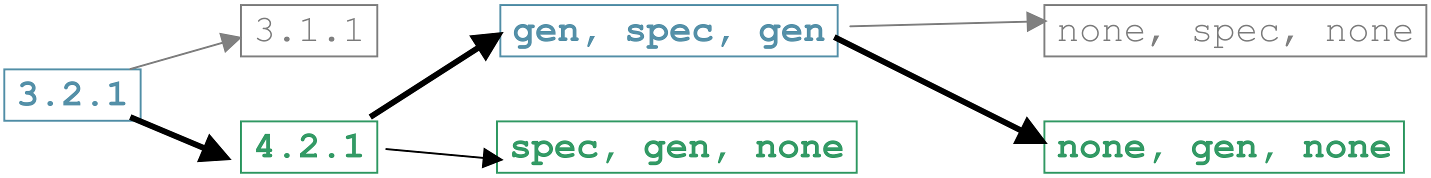
- Metadata:
  - for each trait and category contains pair *(revnum,diff)*

```
<compdata system="sofa">
 <provider>cz.zcu.kiv</provider>
 <name> FAddressBook </name>

 <version>
   <branch>trunk</branch>
   <revision>  <parent>3.1.1</parent>
    <data level="component">
     <trait><name> E </name>
       <revnum> 4 </revnum>
       <change> spec </change></trait>
     <trait><name> N </name>
       <revnum> 2 </revnum>
       <change> spec </change></trait>
     <trait><name> T </name>
       <revnum> 1 </revnum>
       <change> none </change></trait>
    </data>
    <data level="trait">
     <!-- folded -->
    </data>
   </revision>
```

The Process of Upgrade

- Component provider: create metadata
  - write component → run diff against previous → generate rev → store (rev,diff,…) in metadata
  - may compute context for known architectures
- Before upgrade: check using metadata
  - extract metadata → check revision → check for subtyping → check in context



3.1.1

3.2.1

4.2.1

gen, spec, gen

spec, gen, none

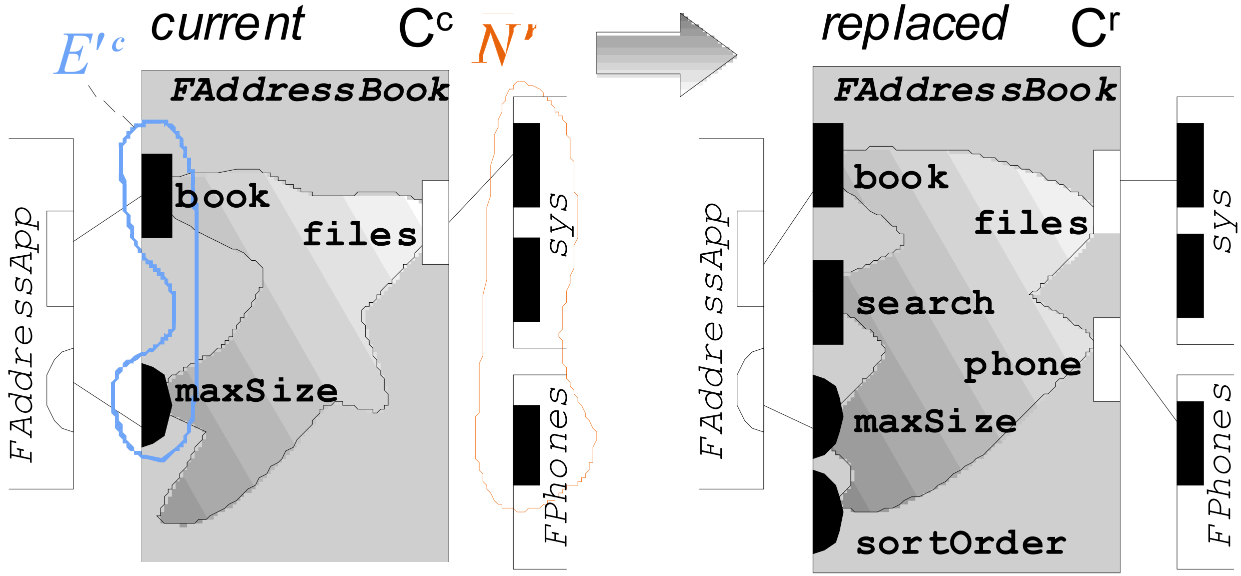
none, spec, none

none, gen, none

diff(C^c,C^r)

diff(Cx,C^r)

Upgrade: Context Considered



```
architecture AdrBook {
  inst app: FAddressApp;
  inst ab: FAddressBook;
  inst p: Fphones;
  set ab.maxSize;
  bind app.book to ab.book;
  bind ab.files to sys.fs;
}
```

- Replacement is allowed to require more, etc.
  - subtyping wrt context
  - $Cx = (E'^c, \overline{N}^r, T^x)$

Conclusions

- Status:
  - concept, work in progress
  - partial implementation of metadata generation
- ✓ Pros:
  - precomputed, precise, automated
  - novel concept of contextual compatibility
- ✗ Cons:
  - limited by data available in specifications, problems with interface adaptation
  - needs practical evaluation