CMDI 1.2
and beyond

Twan Goosen and Menzo Windhouwer
Max Planck Institute for Psycholinguistics

twan.goosen@mpi.nl
menzo.windhouwer@mpi.nl

CMDI 1.2

- Relatively large changes to
  - "General component schema" (for comp specs)
  - Profile schema generation (spec -> xsd)
  - And therefore the structure of instances

- Changes mostly not backwards compatible

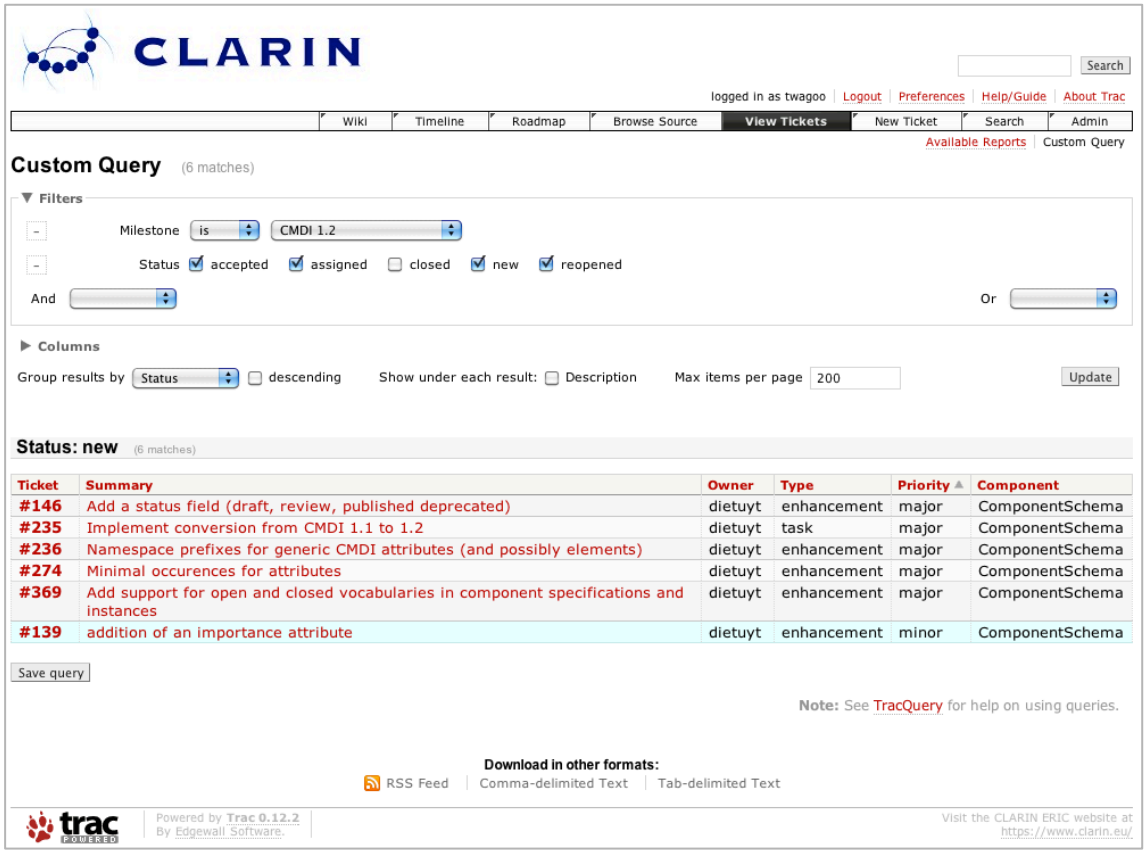
CMDI 1.1 -> CMDI 1.2

- Keep supporting CMDI 1.1
- How will this be managed in Component Registry?
  - CMDI version number in API call, e.g.: *http://catalog.clarin.eu/ComponentRegistry/…**/{profileId}/1.2/xsd***
  - No version specified will imply 1.1 so as not to break existing software
- Existing components
  - Will be upgraded to 1.2
  - 1.1 version remains available through transformation
- New components
  - Will always be 1.2
  - Registry provides *lossy* conversion to 1.1 (lacking 1.2 features)
- Existing metadata
  - Provide transformation to convert 1.1 instances to 1.2 (not vice versa)

CMDI 1.1 -> CMDI 1.2

- Tools
  - Will not break with existing metadata
  - But many tools probably won't check the CMDI version
    - Could lead to issues due to false assumptions about metadata structure when dealing with 1.2
  - New (versions of) tools should check
    - The check is easy to implement (`@CMDVersion`)
    - Supporting both CMDI versions can be done through XML transformation of 1.1 instance to 1.2

TRAC Milestone

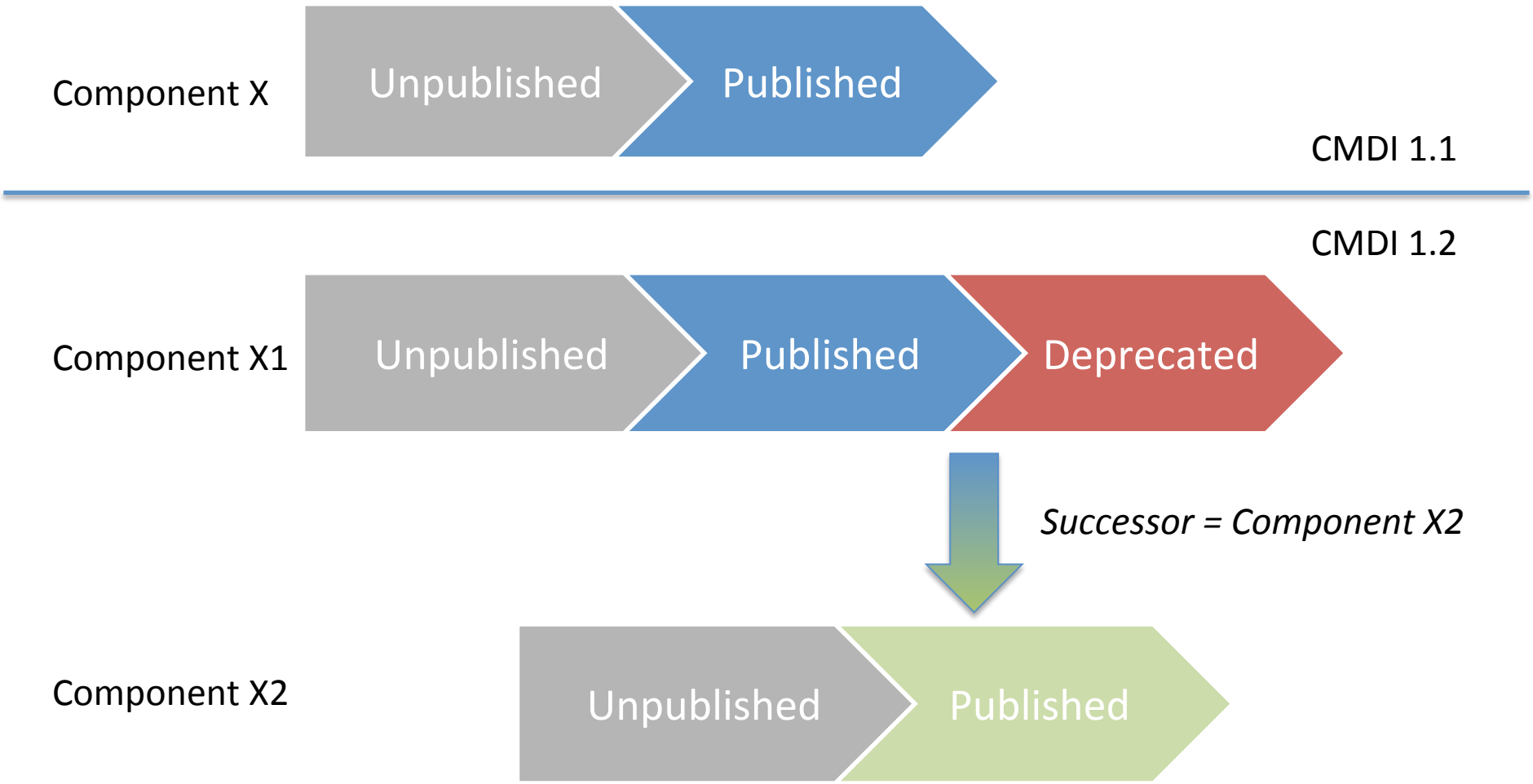


- CMDI 1.2 enhancements and tasks are collected on TRAC
- https://trac.clarin.eu/milestone/CMDI 1.2
- No strict due date, planned late 2013

CMDI 1.2 Changes overview

- Versioning and deprecation
- Attribute issues
- External vocabularies
- Profile reference
- Schema cleanup
- Resource proxy references
- Queues for tools

Versioning and deprecation



Versioning and deprecation

- Components gets a **status header**
  - `<Status>…</Status>`
    - `development`
    - `production`
    - `deprecated` (also for superseded)
- Superseded components have a **successor header**
  - e.g. `<Successor>p_1369140737150</Successor>`
- Deprecated components may get a **comment header**
  - e.g. `<StatusComment>This was deprecated because…`
- To be discussed: also add `<DerivedFrom>` header?

- https://trac.clarin.eu/wiki/CmdiVersioning
- https://trac.clarin.eu/ticket/146

Attribute namespace in instances

- All attributes in instances are in *no-namespace*
  - Component defined (`@myattribute`)
  - CMDI common (`@ref`, `@componentId`)
- Consequence: certain custom attributes are forbidden

- Solution:
  - CMDI common attributes move to namespace, e.g. `@cmd:ref`
  - Component defined attributes stay in no-namespace

- https://trac.clarin.eu/ticket/236
- https://trac.clarin.eu/wiki/CmdiMultipleRecords

Mandatory attributes

- Currently, attributes are always optional
  - Does not allow for mimicking of constraints of existing models
    - E.g. TEI header: profileDesc/langUsage/language/@ident
  - Also simply needlessly restricted

- Solution:
  - optional element `<required>true</required>` in component spec makes attribute required

- https://trac.clarin.eu/ticket/274

External vocabularies (CLAVAS)

- Centralised vocabulary service based on SKOS
  - language codes, organisation names, DCR's, …
- Externalisation allows open, dynamic vocabularies in CMDI
- Closed vocabularies can be imported through Component Registry

- https://trac.clarin.eu/wiki/CmdiClavasIntegration
- https://trac.clarin.eu/ticket/369
- https://trac.clarin.eu/ticket/370

External vocabularies (CLAVAS)

- Vocabulary URI specified in component spec with `@vocabulary`
  -
```
<CMD_Element
 name="organisation"
 vocabulary="http://openskos.org/institutions"
 label="organisation-name" />
```
- Schema inherits these values as `@cmd:vocabulary`, `@cmd:label`
- Vocabulary item URI specified in instance along with text value in `@cmd:ValueConceptLink`
  -
```
<organisation
 cmd:ValueConceptLink="http://openskos.org/api/
institution/beng">
     Sound and Vision
</organisation>
```

Profile reference in instance

- @xsi:schemaLocation is not mandatory
- Header element `<MdProfile>` is optional too
- Therefore, instances can be valid without referring to profile at all!

- CMDI 1.2 makes the `<MdProfile>` header mandatory

- https://trac.clarin.eu/ticket/403

Proposal: schema cleanup

- Usage of elements/attributes in component spec inconsistent
  - e.g. `<CMD_Element ConceptLink="">`
    vs `<Attribute>`
    `<ConceptLink>`
- Resource relation list
  - `<ResourceRelation>`
  - Request to rename `<res1>`, `<res2>` elements
  - Is it used at all?

Proposals: ResourceProxy references

- With namespace prefix: `@cmd:ref`
- Currently multiple reference from one component
  - Blocks resolution of issues with OAI-PMH
  - Proposal: only one ref per component/element
- Currently only on components
  - Component describes resource, elements its aspects
  - But perhaps should also be allow from elements?

Queues for tools (aka AppInfo)

- Currently:
  - *Documentation* and *DisplayPriority* for elements
  - *AppInfo* for display values of CV items
- Possible extensions
  - Importance attribute:
    - deprecated/optional/recommended/mandatory
  - Display structure information (merge components into single node, e.g. *Actor/BirthCountry*)
  - Calculated values (e.g. fill *age* from *date of birth*)
  - Auto creation of technical metadata (e.g. mime type, file size, encoding) in appropriate fields
- Syntax in component spec:
  - Attributes or children of concerning components/elements
  - Possibly separate namespace
- Syntax in XSD:
  - In AppInfo blocks for relatively large chunks
  - Sometimes @ann:… attributes more sensible (compactness)

Beyond CMDI 1.2

Instance versioning
Recursive hierarchies
Internationalisation
Resource reference typing

Metadata instance versioning

- Repositories often want to keep versions of resources and metadata
- Requires a way to link versions
  - In CMDI model `<JournalFileProxy>` is the obvious linking point
  - Proposal by Florian Schiel: https://trac.clarin.eu/wiki/CmdiVersioning
- Standardise syntax for journal file?
- Exclude old versions from OAI?

Recursive hierarchies

- The following *can* be expressed in a component specification:
  - Component A
    - Links to Component B
      - Links to Component A
- But it will cause the transformation to schema to loop forever, and the Component Registry does not allow it
- Transformation *could* be changed to allow infinite depth
- Some tools may work on the assumption that the number of paths in a profile is finite

CMDI internationalisation

- CMDI support localisation on the instance level…
- No localisation at component specification level
  - Localised element descriptions?
  - Localised element and component *display* names?
  - Localised vocabulary display names? (CLAVAS)

ResourceRef typing

- Resource link resource type coherence, guidance and constraints
- Slides by Daan

Overview and status (1)

Feature	Implementation proposal	CMDI 1.2	Beyond 1.2?
Component status and versioning headers	✔	✔	
Component "derivedFrom" header	✔	✔	
Generic CMDI attributes in namespace	✔	✔	
Mandatory attributes	✔	✔	
Link profile elements to external vocabularies	✔	✔	
Make MdProfile header mandatory	✔	✔	
Extended application queues	✔	✔	

Overview and status (2)

Feature	Implementation proposal	CMDI 1.2	Beyond 1.2?
Remove ResourceRelation set	✔	?	
Improve component schema consistency	✔	?	
Limit resource proxy ref to single id	✔	?	
Allow resource proxy ref on component	✔	?	
Instance versioning	✔		?
Allow recursive component hierarchies			?
Component internationalisation			?
ResourceRef typing			?