A proposal for modular XMCDA

Olivier Cailloux

LAMSADE, Université Paris-Dauphine

20th September, 2017

https://github.com/oliviercailloux/XMCDA-Modular-road

My goal (broadly)

- Improve XMCDA
- Long term goal!
- Compatible with evolutions to come (XMCDA v3)
- This is not a criticism of existing work
- *I*'ll (try to) take care of it
- Comments / Help welcome

Outline

1. My goal
2. Overview
3. Current state
4. Modular XMCDA
5. Implementation

Outline

1. My goal
2. Overview
3. Current state
4. Modular XMCDA
5. Implementation

My goal (more specifically)

- We want to let Web Services (WSs) communicate
- Communication requires standard language
- Common language ≠ constraints specific to WS
- Currently: non standard encoding of specific constraints (diviz-like)
- Not taken into account in your programming language
- SOAP calls allowed, but does not embrace standards
- My goal 1: improve compatibility with standards
- My goal 2: ease development
- My constraint: compatibility with XMCDA

Outline

1. My goal
2. Overview
3. Current state
4. Modular XMCDA
5. Implementation

Example of WSs

WS 1

- I want a non-empty list of alternatives each having a name and a value
- Example input (conceptually) : $\{ ("a1", 3), ("a2", 10), ("blah", 4) \}$
- I provide as an output an ordered list of alternatives
- Example output: $("a2", "blah", "a1")$

WS 2

- Input: an ordered list of alternatives; evaluations for each of these alternatives; a value function
- Output: binary (is the list ranked according to the value function?)

Desired

- In example: Common encoding of “ordered list of alternatives”
- In general: Common encoding of other inputs and outputs: “list of alternatives each having a name and a value”, …
- With low redundancy: only one way of specifying each possible input type
- Common language, but each WS has specific inputs and outputs

Current approach

Common language

- Each WS must accept and produce XMCDA
- Specified with a schema
- Version 3 in preparation
- Improvements done about “low redundancy”

Specificities of WSs

- Not every WS accepts every XMCDA valid input
- Each WS specifies its supplementary constraints in a diviz web descriptor
- Diviz uses this for its user interface
- Developers manually check their input

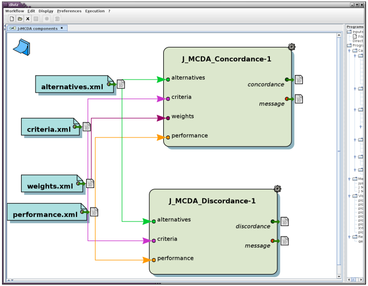
Some details

Common language

- XMCDA schema specifies the grammar of valid XMCDA encoding
- Very permissive (out of necessity): has to accept inputs and outputs that are valid for at least one WS
- Valid iff contains alternatives OR criteria OR value functions OR ...

Specific constraints

- Developer specifies input type
- Through Diviz descriptor (example)



Some details: simplify developers work

Also: simplify developers work

- Diviz framework cuts this input into pieces: alternatives; value functions; …
- Developer programs a WS by simply reading from different inputs (one input per type)

But…

- Tendency to mandate use of library to guarantee constraints satisfaction beyond schema valid files

Outline

1 My goal

2 Overview

3 Current state

4 Modular XMCDA

5 Implementation

Goals: More standard conforming

- Use XML schemas to constrain every WSs
- SOAP calls with schema constrained input
- Automatic discovery of compatible WSs possible
- Automatic invocation and GUI generation possible
- Better integration with enterprise infrastructure
- Decentralisation: provide XMCDA WS with industry standard tools
- Easier interface with existing services (example: solve LP problem)

Goals: Ease developer work

- Give guarantee: input type has been checked against some specific grammar
- Example: I know I receive a list of ...
- Possible to use libraries to transform XML to language specific types

Constraints

- Interface various WSs
- Reuse global encoding standard (XMCDA v2 / v3)
- Compatibility with existing WSs
- As easy to use for non-XML aware developers

Approach

- XMCDA-Modular (XM) Standard provides a set of types
- Called XM types
- Example: set of alternatives, value function, …
- Each WS has a specific schema
- This schema is composed of a list of entries
- Some of them are XM types

Compare to: one schema for every WS

Reuse

- The single types could be borrowed from XMCDA v. x
- Some types could even be made more constrained
- Developer may provide its own type when too specific for deserving being in the standard
- Developers may re-use types from the industry, e.g. LP
- Compatibility with existing WSs: have to develop translators (per type)

Developer POV

- Instead of Diviz descriptor, has to provide a schema
- More difficult!
- Possible to help with tools (to be developed!)
- Possible conversion with Diviz descriptor files
- Leverage existing tools for easy / automatic parsing of input / output

⇒ To be developed as well!

- Provides standard, more general functionality equivalent to existing manually programmed libraries, with support for languages to come

Infrastructure

- Leverage WSDL, language for SOAP WS description
- Provides standard, more general equivalent to Diviz descriptor
- Extend Diviz to account for standard (WSDL) descriptors suppl to Diviz-like

Outline

1. My goal
2. Overview
3. Current state
4. Modular XMCDA
5. Implementation

Implementation

- Course at Dauphine, MIAGE M2
- Some of these students know the prerequisites
- Could be interested in (parts of) this as student projects

Proposed course of action

- You comment
- We discuss
- I (try to) get parts of the idea implemented
- Using XMCDA-Modular schema types to get started
- To be developed progressively!
- I keep you posted
- Possibly: request some resources for online hosting (later!)
- This proposal is compatible with any improvements to the XMCDA schema
- Don't hold your breath!

Thank you for your attention!