The challenge of getting useful MCDA methods being used through usable software

Tommi Tervonen

Faculty of Economics and Business
University of Groningen
The Netherlands
Introduction

- Use of MCDA methods hasn’t increased substantially during the past 20 years
- Amount of papers (gScholar) mentioning “multi-criteria decision”
  - Pre-1999: 4730
  - Post-1999: about 18600

- MCDA Methods have scientifically been proven to be useful, but not used because of ... ???
>... not being usable due to lack of software?

>... not being used due to incompatibility of existing software with the rest of the decision process?

Decision Deck to the rescue!
MCDA software user groups (from DD)

- **Practitioners** need easy-to-use software with advanced visualization, strict parameter checking, and hiding of unnecessary technical parameters

- **Teachers** need all model parameters to be visible and possibility to disable checking of feasible parameter ranges

- **Researchers** need possibility to extends the methods and to compare results of different methods
But how can these requirements possibly be fulfilled by DD or other MCDA software?

I will shortly analyze three cases:

1. JSMAA
2. JSMAA integrated into ADDIS
3. JSMAA DD adapter
Case 1: JSMAA

- Target user group: Researchers & Practitioners

- Minimizes user interaction (automatic scale computation, multi-threaded simulation)
How to get customers?

- Practitioners
  - Target academic practitioners with real problems outside the core fields of MCDA
  - “It’s free”

- Researchers
  - Open source with:
    1) github
    2) nightly builds
  - Developed in agile manner => can be refactored for integration with an existing ICT infrastructure
Case 2: JSMAA ADDIS integration

ADDIS
- Storage and meta-analysis of aggregate clinical data

ADDIS + JSMAA
- (Semi) automatic model construction
Case 3: JSMAA Decision Deck adapter

- XMCDA enables interoperable services
- Diviz allows to compose workflows of independent web services
- But why do I have to work with files in the 21st century?
- Why aren’t independent SOAP web services supported as XMCDA enables that? (instead of using SOAP marshalling I have to parse input)
>Diviz architecture
> What happens here?

1. Diviz **server** calls a web service (WS) somewhere
2. WS receives input files
3. WS stores the files in a directory
4. WS calls service program (SP)
5. SP loads files in input directory
6. SP parses XML into data structures
7. SP computes, and produces output data structures
8. SP parses output XML from output data structures
9. SP stores output XML files in the output directory
10. WS loads the output files and transmits them to Diviz
How it would be with standard web services

1. Diviz client calls a web service somewhere. Web services are discoverable, but also the central server can work as a service locator.

2. Diviz calls web service method() with POJO Bean parameters, that are automatically marshaled for XMCDA. No user code is involved until the actual web service method() is called.

3. method() returns an Object, that is automatically marshaled into an XMCDA reply.

4. The rest is taken care of by middleware.
Advantages of changing architecture

- Standard way of doing things, all required software already exists open source
- De-centralization. Allows DD to be extended freely without control of the DD consortium
- Direct use of XMCDA with SOAP marshalling
- Real cloud computing
Disadvantages of changing architecture

- Additional technical knowledge required to make “real” web services
- XMCDA-SOAP translation required
- SOAP marshaling is not readily available for all programming languages (e.g. PHP, Python)
- Lose centralized control (?)
Conclusions

- XMCDA enables SOA in ways not yet fully exploited in DD (cloud computing)
- In order to satisfy requirements of all stated user groups, DD needs a more loosely coupled architecture
- Needs of practitioners cannot be satisfied by Diviz, because they need a more refined UI and integration with existing ICT infrastructure

> [www.smaa.fi](http://www.smaa.fi)
> [github.com/tommite/jsmaa](http://github.com/tommite/jsmaa)