Package ‘RXMCDA’

December 11, 2015

Version 1.5.5
Date 2015-11-16
Title Functions to Parse and Create XMCDA Files
Author Patrick Meyer and Sebastien Bigaret, with contributions from Krzysztof Ciomek.
Maintainer Patrick Meyer <patrick.meyer@telecom-bretagne.eu>
Description Functions which allow to read many XMCDA tags and transform them into R variables which are then usable in MCDA algorithms written in R. It also allows to write certain R variables into XML files respecting the XMCDA standard.
Depends XML
Imports kappalab
Encoding latin1
License CeCILL-2
URL https://github.com/paterijk/RXMCDA
NeedsCompilation no
Repository CRAN
Date/Publication 2015-12-11 16:50:02

R topics documented:

checkXSD ......................................................... 2
getAlternativesAffectations .................................. 3
getAlternativesComparisons .................................. 4
getAlternativesComparisonsLabels .......................... 6
getAlternativesComparisonsValues .......................... 7
getAlternativesIDs ............................................. 8
getAlternativesIntervalValues ............................... 10
getAlternativesValues ........................................ 11
getCategoriesIDs .............................................. 12
getCategoriesIntervalValues ................................. 13
getCategoriesValues ......................................... 14
getCriteriaComparisons ...................................... 15
Description

Checks if an XML tree is valid according to an XML schema.

Usage

checkXSD(tree)
getAlternativesAffectations

Arguments

tree Object containing the XMCDA XML tree.

Value

The function returns 1 if the XML tree is validated by the XMCDA schema (currently XMCDA-2.*), else it returns 0.

Examples

tree = newXMLDoc()

newXMLNode("xmcda:XMCDA",
    namespace = c("xsi" = "http://www.w3.org/2001/XMLSchema-instance",
    "xmcda" = "http://www.decision-deck.org/2009/XMCDA-2.1.0"),
    parent=tree)

root<-getNodeSet(tree, "/xmcda:XMCDA")

criteria<-newXMLNode("criteria", parent=root[[1]], namespace=c())

criterion<-newXMLNode("criterion", attrs = c(id="g1"),
    parent=criteria, namespace=c())

y<-checkXSD(tree)

getAlternativesAffectations

Get alternative affectations

Description

Gets alternative affectations, stored in the <alternativesAffectations> tag, from an XML tree written according to the XMCDA standard.

Usage

getAlternativesAffectations(tree, alternativesIDs, categoriesIDs, mcdaConcept = NULL)

Arguments

tree Object containing the XMCDA XML tree.

alternativesIDs A vector containing the IDs of the alternatives to be considered for the extractions.

categoriesIDs A vector containing the IDs of the categories to be considered for the extractions.

mcdaConcept A string containing the specific mcdaConcept attribute which should be searched for.
getAlternativesComparisons

Value

The function returns a list structured as follows:

\[
\begin{align*}
\text{The first elements contain } n \times m \text{ matrices representing each an } <\text{alternativesAffectations}>, \\
\text{where } n \text{ is number of considered alternatives and } m \text{ is number of considered categories. Each cell } [i, j] \text{ of each matrix is boolean and corresponds to encoded relation between } i\text{-th alternative and } j\text{-th category. Elements of returned list are named according to the mcdaConcept attribute if it can be found.}
\end{align*}
\]

status

Either OK if all the \(<\text{alternativesAffectations}>\) tags could be correctly read, or the description of the error.

Examples

```r

tree <- xmlTreeParse(system.file("extdata",
   "testFile.xml",
   package="RXMCDA"),
   useInternalNodes=TRUE)

alternativesIDs <- getAlternativesIDs(tree)
categoriesIDs <- getCategoriesIDs(tree)
altAff <- getAlternativesAffectations(tree,
   alternativesIDs[[1]], categoriesIDs[[1]])
```

Description

Extracts \(<\text{alternativesComparisons}>\) from an XML tree written according to the XMCDA standard and stores the performances of the concerned alternatives.

Usage

```r
getAlternativesComparisons(tree, performanceTable, mcdaConcept = NULL)
```

Arguments

tree

Object containing the XMCDA XML tree.

performanceTable

A matrix containing the performance table to be considered, whose dimnames are the alternatives’ IDs (rows) and the criteria’s IDs (columns).

mcdaConcept

A string containing the specific mcdaConcept attribute which should be searched for.
getAlternativesComparisons

Value

The function returns a list structured as follows:

- The first elements contain the <alternativesComparisons> found in <tree> as matrices. These elements are named according to the mcdaConcept attribute if it can be found. Each line of each matrix corresponds to one constraint of the type "alternative a is preferred to alternative b with preference threshold delta". A line is structured as follows: the first n elements encode alternative a (via the values it takes in performanceTable), the next n elements encode alternative b, and the last element contains the preference threshold delta.

status

Either OK if all the <criteria> tags could be correctly read, or the description of the error.

Examples

tree = newXMLDoc()

newXMLNode("xmcda:XMCDA",
    namespace = c("xsi" = "http://www.w3.org/2001/XMLSchema-instance",
    "xmcda" = "http://www.decision-deck.org/2009/XMCDA-2.1.0"),
    parent=tree)

root<-getNodeSet(tree, "/xmcda:XMCDA")

aC<-newXMLNode("alternativesComparisons", parent=root[[1]], namespace=c())

pairs<-newXMLNode("pairs", parent=aC, namespace=c())
pair<-newXMLNode("pair", parent=pairs, namespace=c())

initial<-newXMLNode("initial", parent=pair)
newXMLNode("alternativeID", "a01", parent=initial, namespace=c())

terminal<-newXMLNode("terminal", parent=pair, namespace=c())
newXMLNode("alternativeID", "a02", parent=terminal, namespace=c())

value<-newXMLNode("value", parent=pair, namespace=c())
newXMLNode("real", "0.5", parent=value, namespace=c())

alternativesIDs <- c("a01","a02")
criteriaIDs <- c("g1","g2")

performance.table<-matrix(nrow=length(alternativesIDs),
    ncol=length(criteriaIDs),
    dimnames = list(alternativesIDs,criteriaIDs))

for (i in 1:length(alternativesIDs)){
    for (j in 1:length(criteriaIDs)){
        performance.table[1,j] = runif(1)
    }
}
getAlternativesComparisonsLabels

Get comparisons of alternatives

Description

Extracts `<alternativesComparisons>` from an XML tree written according to the XMCDA standard and stores the IDs of the concerned alternatives.

Usage

`getAlternativesComparisonsLabels(tree, altIDs=NULL, mcdaConcept = NULL)`

Arguments

- `tree` Object containing the XMCDA XML tree.
- `altIDs` A vector containing the IDs of the alternatives to be considered for the extractions.
- `mcdaConcept` A string containing the specific `mcdaConcept` attribute which should be searched for.

Value

The function returns a list structured as follows:

```
- The first elements contain the `<alternativesComparisons>` found in `<tree>` as matrices. These elements are named according to the `mcdaConcept` attribute if it can be found. Each line of each matrix corresponds to one constraint of the type "alternative a is preferred to alternative b with preference threshold delta". A line is structured as follows: the first element encodes the ID of alternative a, the second element encodes the ID of alternative b, and the last element contains the preference threshold delta.
- status Either OK if all the `<alternativesComparisons>` tags could be correctly read, or the description of the error.
```

Examples

```r
tree = newXMLDoc()

newXMLNode("xmcda:XMCDA",
  namespace = c("xsi" = "http://www.w3.org/2001/XMLSchema-instance",
  "xmcda" = "http://www.decision-deck.org/2009/XMCDA-2.1.0"),
  parent=tree)
```

```r
ty<-getNodeSet(tree,"//alternativesComparisons")

x<-getAlternativesComparisons(y[[1]], performance.table)
```
getAlternativesComparisonsValues

Get alternatives comparisons values

Description

Gets alternatives comparisons values stored in the <alternativesComparisons> tag, from an XML tree written according to the XMCDA standard.

Usage

getAlternativesComparisonsValues(tree, alternativesIDs, mcdaConcept = NULL)

Arguments

- tree: Object containing the XMCDA XML tree.
- alternativesIDs: A vector containing the IDs of the alternatives to be considered for the extractions.
- mcdaConcept: A string containing the specific mcdaConcept attribute which should be searched for.

Value

The function returns a list structured as follows:

```r
root <- getNodeSet(tree, "/xmcda:XMCDA")

aC <- newXMLNode("alternativesComparisons", parent=root[[1]], namespace=c())
pairs <- newXMLNode("pairs", parent=aC, namespace=c())
pair <- newXMLNode("pair", parent=pairs, namespace=c())

initial <- newXMLNode("initial", parent=pair, namespace=c())
newXMLNode("alternativeID", "a01", parent=initial, namespace=c())

terminal <- newXMLNode("terminal", parent=pair, namespace=c())
newXMLNode("alternativeID", "a02", parent=terminal, namespace=c())

value <- newXMLNode("value", parent=pair, namespace=c())
newXMLNode("real", "0.5", parent=value, namespace=c())

alternativesIDs <- c("a01","a02")

y <- getNodeSet(tree,"//alternativesComparisons")
x <- getAlternativesComparisonsLabels(y[[1]], alternativesIDs)
```
getAlternativesIDs

---

The first elements contain matrices representing each an `<alternativesComparisons>`. Each row \( c(i, j, v) \) of each matrix corresponds to a comparison between \( i \)-th `<initial>` and \( j \)-th `<terminal>` alternatives with value \( v \). Elements of returned list are named according to the `mcdaConcept` attribute if it can be found.

**status**

Either OK if all the `<alternativesComparisons>` tags could be correctly read, or the description of the error.

**Examples**

```plaintext

tree = newXMLDoc()

newXMLNode("xmcda:XMCDA",
    namespace = c("xsi" = "http://www.w3.org/2001/XMLSchema-instance",
    "xmcda" = "http://www.decision-deck.org/2009/XMCDA-2.1.0"),
    parent=tree)

root <- getNodeSet(tree, "/xmcda:XMCDA")
altComp <- newXMLNode("alternativesComparisons", parent=root[[1]], namespace=c())
pairs <- newXMLNode("pairs", parent=altComp, namespace=c())

pair <- newXMLNode("pair", parent=pairs, namespace=c())
initial <- newXMLNode("initial", parent=pair)
newXMLNode("alternativeID", "a01", parent=initial, namespace=c())
terminal <- newXMLNode("terminal", parent=pair, namespace=c())
newXMLNode("alternativeID", "a02", parent=terminal, namespace=c())
value <- newXMLNode("value", parent=pair, namespace=c())
newXMLNode("real", "1", parent=value, namespace=c())

pair <- newXMLNode("pair", parent=pairs, namespace=c())
initial <- newXMLNode("initial", parent=pair)
newXMLNode("alternativeID", "a01", parent=initial, namespace=c())
terminal <- newXMLNode("terminal", parent=pair, namespace=c())
newXMLNode("alternativeID", "a03", parent=terminal, namespace=c())
value <- newXMLNode("value", parent=pair, namespace=c())
newXMLNode("real", "9", parent=value, namespace=c())

alternativesIDs <- c("a01", "a02", "a03")
x <- getAlternativesComparisonsValues(tree, alternativesIDs)
```

---

**getAlternativesIDs**

Get IDs of alternatives

**Description**

Gets the IDs of alternatives, stored in the `<alternatives>` tag, from an XML tree written according to the XMCDA standard.
**getAlternativesIDs**

Usage

```c
getAlternativesIDs(tree, mcdaConcept = NULL)
```

Arguments

- **tree**: Object containing the XMCDA XML tree.
- **mcdaConcept**: A string containing the specific mcdaConcept attribute which should be searched for.

Value

The function returns a list structured as follows:

- **--**: The first elements contain vectors with the alternatives’ IDs which have been found in each `<alternatives>` tag. These elements are named according to the mcdaConcept attribute if it can be found.
- **status**: Either OK if all the `<alternatives>` tags could be correctly read, or the description of the error.

Examples

```c
tree = newXMLDoc()

newXMLNode("xmcda:XMCDA",
    namespace = c("xsi" = "http://www.w3.org/2001/XMLSchema-instance",
                   "xmcda" = "http://www.decision-deck.org/2009/XMCDA-2.0.0"),
    parent = tree)

root<-getNodeSet(tree, "/xmcda:XMCDA")

alternatives<-newXMLNode("alternatives", attrs=c(mcdaConcept="actions"),
        parent=root[[1]],
        namespace=c())

alternative<-newXMLNode("alternative", attrs = c(id="x1"),
                    parent=alternatives, namespace=c())
alternative<-newXMLNode("alternative", attrs = c(id="x2"),
                    parent=alternatives, namespace=c())
alternative<-newXMLNode("alternative", attrs = c(id="x3"),
                    parent=alternatives, namespace=c())

y<-getNodeSet(tree,"//alternatives")
x<-getAlternativesIDs(y[[1]])
```
getAlternativesIntervalValues

Get interval values related to alternatives

Description

Gets intervals of values related to alternatives, stored in the \texttt{<alternativesValues>} tag, from an XML tree written according to the XMCDA standard.

Usage

\texttt{getAlternativesIntervalValues(tree, alternativesIDs, mcdaConcept = NULL)}

Arguments

- **tree**: Object containing the XMCDA XML tree.
- **alternativesIDs**: A vector containing the IDs of the alternatives to be considered for the extractions.
- **mcdaConcept**: A string containing the specific mcdaConcept attribute which should be searched for.

Value

The function returns a list structured as follows:

\begin{verbatim}
-- The first elements contain matrices representing each an \texttt{<alternativesValues>}. Each line of each matrix corresponds to an element of the type "the interval of values assigned to alternative a is [x,y]". A line is structured as follows: the first element encodes the index of the ID of alternative a in \texttt{alternativesIDs}, the second element encodes the value x, and the last element encodes the value y. These elements are named according to the mcdaConcept attribute if it can be found.

status Either OK if all the \texttt{<alternativesValues>} tags could be correctly read, or the description of the error.
\end{verbatim}

Examples

\begin{verbatim}
tree <- xmlTreeParse(system.file("extdata",
  "testFile.xml",
  package="RXMLCDA"),
  useInternalNodes=TRUE)

altIDs <- getAlternativesIDs(tree)

altVals <- getAlternativesIntervalValues(tree, altIDs[[1]],
  mcdaConcept="alternativesIntervalValues")
\end{verbatim}
getAlternativesValues  Get values related to alternatives

Description

Gets values related to alternatives, stored in the <alternativesValues> tag, from an XML tree written according to the XMCDA standard.

Usage

getAlternativesValues(tree, alternativesIDs, mcdaConcept = NULL)

Arguments

tree Object containing the XMCDA XML tree.
alternativesIDs A vector containing the IDs of the alternatives to be considered for the extrac-
tions.
mcdaConcept A string containing the specific mcdaConcept attribute which should be searched
for.

Value

The function returns a list structured as follows:

--- The first elements contain matrices representing each an <alternativesValues>. Each line of each matrix corresponds to an element of the type "the value assigned to alternative a is x". A line is structured as follows: the first element encodes the index of the ID of alternative a in alternativesIDs, the second element encodes the value x. These elements are named according to the mcdaConcept attribute if it can be found.

status Either OK if all the <alternativesValues> tags could be correctly read, or the
description of the error.

Examples

tree <- xmlTreeParse(system.file("extdata",
    "testFile.xml",
    package="RXMCDA"),
    useInternalNodes=TRUE)

altIDs <- getAlternativesIDs(tree)

altVals <- getAlternativesValues(tree, altIDs[[1]])
**getCategoriesIDs**  
*Get IDs of categories*

**Description**

Gets the IDs of categories, stored in the `<categories>` tag, from an XML tree written according to the XMCDA standard.

**Usage**

```
getCategoriesIDs(tree, mcdaConcept = NULL)
```

**Arguments**

- **tree**: Object containing the XMCDA XML tree.
- **mcdaConcept**: A string containing the specific mcdaConcept attribute which should be searched for.

**Value**

The function returns a list structured as follows:

- **--**: The first elements contain vectors with the categories’ IDs which have been found in each `<categories>` tag. These elements are named according to the mcdaConcept attribute if it can be found.
- **status**: Either OK if all the `<categories>` tags could be correctly read, or the description of the error.

**Examples**

```r
tree = newXMLDoc()

newXMLNode("xmcda:XMCDA",
    namespace = c("xsi" = "http://www.w3.org/2001/XMLSchema-instance",
                 "xmcda" = "http://www.decision-deck.org/2009/XMCDA-2.0.0"),
    parent=tree)

root<-getNodeSet(tree, "/xmcda:XMCDA")

categories<-newXMLNode("categories", attrs=c(mcdaConcept="classes"),
    parent=root[[1]],
    namespace=c())

newXMLNode("category", attrs = c(id="c1"), parent=categories, namespace=c())
newXMLNode("category", attrs = c(id="c2"), parent=categories, namespace=c())
newXMLNode("category", attrs = c(id="c3"), parent=categories, namespace=c())

y<-getNodeSet(tree,"//categories")
x<-getCategoriesIDs(y[[1]])
```
getCategoriesIntervalValues

Get interval values related to categories

Description

Gets interval values related to categories, stored in the `<categoriesValues>` tag, from an XML tree written according to the XMCDA standard.

Usage

getCategoriesIntervalValues(tree, categoriesIDs, mcdaConcept = NULL)

Arguments

tree Object containing the XMCDA XML tree.
categoriesIDs A vector containing the IDs of the categories to be considered for the extractions.
mcdaConcept A string containing the specific mcdaConcept attribute which should be searched for.

Value

The function returns a list structured as follows:

- The first elements contain matrices representing each an `<categoriesValues>`. Each line of each matrix corresponds to an element of the type "the interval value assigned to category c1 is [x, y]". A line is structured as follows: the first element encodes the index of the ID of category c1 in categoriesIDs, the second element encodes the value x and the third element encodes the value y. These elements are named according to the mcdaConcept attribute if it can be found.

status Either OK if all the `<categoriesValues>` tags could be correctly read, or the description of the error.

Examples

tree <- xmlTreeParse(system.file("extdata",
  "testFile.xml",
  package="RXMCDA"),
  useInternalNodes=TRUE)

categoriesIDs <- getCategoriesIDs(tree)
intervalValues <- getCategoriesIntervalValues(tree, categoriesIDs[[1]])
getCategoriesValues  Get values related to categories

Description

Gets values related to categories, stored in the <categoriesValues> tag, from an XML tree written according to the XMCDA standard.

Usage

getCategoriesValues(tree, categoriesIDs, mcdaConcept = NULL)

Arguments

tree  Object containing the XMCDA XML tree.
categoriesIDs  A vector containing the IDs of the categories to be considered for the extractions.
mcdaConcept  A string containing the specific mcdaConcept attribute which should be searched for.

Value

The function returns a list structured as follows:

- The first elements contain matrices representing each an <categoriesValues>. Each line of each matrix corresponds to an element of the type "the value assigned to category c1 is x". A line is structured as follows: the first element encodes the index of the ID of category c1 in categoriesIDs, the second element encodes the value x. These elements are named according to the mcdaConcept attribute if it can be found.

status  Either OK if all the <categoriesValues> tags could be correctly read, or the description of the error.

Examples

tree <- xmlTreeParse(system.file("extdata",
  "testFile.xml",
  package="RXMCDA"),
  useInternalNodes=TRUE)
categoriesIDs <- getCategoriesIDs(tree)
categoriesValues <- getCategoriesValues(tree, categoriesIDs[[1]])
getCriteriaComparisons

Get comparisons of criteria

Description

Gets comparisons of criteria, stored in the <criteriaComparisons> tag, from an XML tree written according to the XMCDA standard.

Usage

getCriteriaComparisons(tree, criteriaIDs, mcdaConcept = NULL)

Arguments

tree Object containing the XMCDA XML tree.
criteriaIDs A vector containing the IDs of the criteria to be considered for the extractions.
mcdaConcept A string containing the specific mcdaConcept attribute which should be searched for.

Value

The function returns a list structured as follows:

-- The first elements contain the <criteriaComparisons> found in <tree> as matrices. These elements are named according to the mcdaConcept attribute if it can be found. Each line of each matrix corresponds to one constraint of the type "criterion g1 is preferred to criterion g2 with preference threshold delta". A line is structured as follows: the first element encodes the index of criterion g1 in criteriaIDs, the next element encodes the ID of criterion g2, and the last element contains the preference threshold delta.

status Either OK if all the <criteriaComparisons> tags could be correctly read, or the description of the error.

Examples

tree <- xmlTreeParse(system.file("extdata",
    "testFile.xml",
    package="RXMCDA"),
    useInternalNodes=TRUE)

critIDs <- getCriteriaIDs(tree)

combs <- getCriteriaComparisons(tree, critIDs[[1]])
getCriteriaComparisonsLabels

*Get comparisons of alternatives*

**Description**

Extracts `<criteriacomparisons>` from an XML tree written according to the XMCDA standard and stores the IDs of the concerned criteria.

**Usage**

`getCriteriaComparisonsLabels(tree, critIDs=NULL, mcdaConcept = NULL)`

**Arguments**

- **tree**
  - Object containing the XMCDA XML tree.
- **critIDs**
  - A vector containing the IDs of the criteria to be considered for the extractions.
- **mcdaConcept**
  - A string containing the specific mcdaConcept attribute which should be searched for.

**Value**

The function returns a list structured as follows:

```
-- The first elements contain the <alternativesComparisons> found in <tree> as matrices. These elements are named according to the mcdaConcept attribute if it can be found. Each line of each matrix corresponds to one constraint of the type "criterion g1 is preferred to criterion g2 with preference threshold delta". A line is structured as follows: the first element encodes the ID of criterion g1, the second element encodes the ID of criterion g2, and the last element contains the preference threshold delta.
```

- **status**
  - Either OK if all the `<criteriacomparisons>` tags could be correctly read, or the description of the error.

**Examples**

```
tree <- xmlTreeParse(system.file("extdata",
"testFile.xml",
package="RXMCDA"),
useInternalNodes=TRUE)
critIDs <- getCriteriaIDs(tree)
comps <- getCriteriaComparisonsLabels(tree, critIDs[[1]])
```
**getCriteriaIDs**  

**Get IDs of criteria**

**Description**

Gets the IDs of criteria, stored in the `<criteria>` tag, from an XML tree written according to the XMCDA standard.

**Usage**

```c
getCriteriaIDs(tree, mcdaConcept = NULL)
```

**Arguments**

- **tree**  
  Object containing the XMCDA XML tree.

- **mcdaConcept**  
  A string containing the specific mcdaConcept attribute which should be searched for.

**Value**

The function returns a list structured as follows:

- The first elements contain vectors with the criteria's IDs which have been found in each `<criteria>` tag. These elements are named according to the mcdaConcept attribute if it can be found.

- **status**  
  Either OK if all the `<criteria>` tags could be correctly read, or the description of the error.

**Examples**

```c
tree = newXMLDoc()
	newXMLNode("xmcda:XMCDA",
		namespace = c("xsi" = "http://www.w3.org/2001/XMLSchema-instance",
		"xmcda" = "http://www.decision-deck.org/2009/XMCDA-2.0.0"),
		parent=tree)

root<-getNodeSet(tree, "/xmcda:XMCDA")

criteria<-newXMLNode("criteria", parent=tree[[1]], namespace=c())
	newXMLNode("criterion",attrs = c(id="g1"), parent=criteria, namespace=c())
	newXMLNode("criterion",attrs = c(id="g2"), parent=criteria, namespace=c())

y<-getNodeSet(tree,"/criteria")

x<-getCriteriaIDs(y[[1]])
```
getCriteriaIntervalValues

*Get interval values related to criteria*

**Description**

Extracts the number of criteria from an XML tree written according to the XMCDA standard.

**Usage**

```r
getCriteriaIntervalValues(tree, criteriaIDs, mcdaConcept = NULL)
```

**Arguments**

- `tree`: Object containing the XMCDA XML tree.
- `criteriaIDs`: A vector containing the IDs of the criteria to be considered for the extractions.
- `mcdaConcept`: A string containing the specific mcdaConcept attribute which should be searched for.

**Value**

The function returns a list structured as follows:

- `--`: The first elements contain the number of criteria of each `<criteria>` found in `<tree>`. These elements are named according to the `mcdaConcept` attribute if it can be found.
- `status`: Either OK if all the `<criteria>` tags could be correctly read, or the description of the error.

**Examples**

```r
tree <- xmlTreeParse(system.file("extdata", "testFile.xml", package="RXMCDA"), useInternalNodes=TRUE)

critIDs <- getCriteriaIDs(tree)

intVals <- getCriteriaIntervalValues(tree, critIDs[[1]])
```
**getCriteriaPairsComparisons**

*Get comparisons of pairs of criteria*

**Description**

Gets comparisons of pairs of criteria, stored in the `<criteriacomparisons>` tag (pairs are represented as sets of two elements), from an XML tree written according to the XMCDA standard.

**Usage**

`getCriteriaPairsComparisons(tree, criteriaIDs, mcdaConcept = NULL)`

**Arguments**

- `tree`: Object containing the XMCDA XML tree.
- `criteriaIDs`: A vector containing the IDs of the criteria to be considered for the extractions.
- `mcdaConcept`: A string containing the specific mcdaConcept attribute which should be searched for.

**Value**

The function returns a list structured as follows:

```
# The first elements contain the <criteriacomparisons> found in <tree> as matrices. These elements are named according to the mcdaConcept attribute if it can be found. Each line of each matrix corresponds to one constraint of the type "criterion g1 is preferred to criterion g2 with preference threshold delta". A line is structured as follows: the first element encodes the index of criterion g1 in criteriaIDs, the next element encodes the ID of criterion g2, and the last element contains the preference threshold delta.
```

- `status`: Either OK if all the `<criteriacomparisons>` tags could be correctly read, or the description of the error.

**Examples**

```
tree <- xmlTreeParse(system.file("extdata", 
    "testFile.xml", 
    package="RXMCDA"), 
    useInternalNodes=TRUE)

critIDs <- getCriteriaIDs(tree)

comps <- getCriteriaPairsComparisons(tree, critIDs[[1]])
```
getCriteriaPairsIntervalValues

Get interval values related to pairs of criteria

Description

Gets interval values related to pairs of criteria, stored in the <criteriavalues> tag (pairs are represented as sets of two elements), from an XML tree written according to the XMCDA standard.

Usage

getCriteriaPairsIntervalValues(tree, criteriaIDs, mcdaConcept = NULL)

Arguments

tree                          Object containing the XMCDA XML tree.
criteriaIDs                  A vector containing the IDs of the criteria to be considered for the extractions.
mcdaConcept                  A string containing the specific mcdaConcept attribute which should be searched for.

Value

The function returns a list structured as follows:

--

The first elements contain the <criteriavalues> found in <tree> as matrices. These elements are named according to the mcdaConcept attribute if it can be found. Each line of each matrix corresponds to one constraint of the type "the interval of values assigned to the couple of criteria (g1,g2) is [x,y]". A line is structured as follows: the first element encodes the index of criterion g1 in criteriaIDs, the next element encodes the ID of criterion g2, and the last two elements contain x and y.

status                       Either OK if all the <criteriavalues> tags could be correctly read, or the description of the error.

Examples

tree <- xmlTreeParse(system.file("extdata",
                                  "testFile.xml",
                                  package="RXMCDA"),
                  useInternalNodes=TRUE)

critIDs <- getCriteriaIDs(tree)

pairsVals <- getCriteriaPairsIntervalValues(tree, critIDs[[1]],
                                            mcdaConcept="interactionIntervals")
getCriteriaPairsValues

*Get values related to pairs of criteria*

**Description**

Gets values related to pairs of criteria, stored in the `<criteriaValues>` tag (pairs are represented as sets of two elements), from an XML tree written according to the XMCDA standard.

**Usage**

`getCriteriaPairsValues(tree, criteriaIDs, mcdaConcept = NULL)`

**Arguments**

- **tree**: Object containing the XMCDA XML tree.
- **criteriaIDs**: A vector containing the IDs of the criteria to be considered for the extractions.
- **mcdaConcept**: A string containing the specific mcdaConcept attribute which should be searched for.

**Value**

The function returns a list structured as follows:

- The first elements contain the `<criteriaValues>` found in `<tree>` as matrices. These elements are named according to the mcdaConcept attribute if it can be found. Each line of each matrix corresponds to one statement of the type "the value assigned to the couple of criteria (g1,g2) is x". A line is structured as follows: the first element encodes the index of criterion g1 in criteriaIDs, the next element encodes the ID of criterion g2, and the last elements contain x.
- **status**: Either OK if all the `<criteriaValues>` tags could be correctly read, or the description of the error.

**Examples**

```r
# Example code

tree <- xmlTreeParse(system.file("extdata",
        "testFile.xml",
        package="RXMLCDA"),
        useInternalNodes=TRUE)

critIDs <- getCriteriaIDs(tree)
pairsVals <- getCriteriaPairsValues(tree, critIDs[[1]],
        mcdaConcept="interactionValues")
```
getCriteriaValues Get values related to criteria

Description

Gets values related to criteria, stored in the <criteriavalues> tag, from an XML tree written according to the XMCDA standard.

Usage

getCriteriaValues(tree, criteriaIDs, mcdaConcept = NULL)

Arguments

tree Object containing the XMCDA XML tree.

criteriaIDs A vector containing the IDs of the criteria to be considered for the extractions.

mcdaConcept A string containing the specific mcdaConcept attribute which should be searched for.

Value

The function returns a list structured as follows:

---

The first elements contain matrices representing each an <criteriavalues>. Each line of each matrix corresponds to an element of the type "the value assigned to criterion gi is x". A line is structured as follows: the first element encodes the index of the ID of criterion gi in criteriaIDs, the second element encodes the value x. These elements are named according to the mcdaConcept attribute if it can be found.

status Either OK if all the <criteriavalues> tags could be correctly read, or the description of the error.

Examples

tree <- xmlTreeParse(system.file("extdata",
   "testFile.xml",
   package="RXMLCDA"),
   useInternalNodes=TRUE)

critIDs <- getCriteriaIDs(tree)

critVals <- getCriteriaValues(tree, critIDs[[1]])
getMobiusCapacities

Description

Gets Mobius capacities, stored in the <criteriaValues> tag, from an XML tree written according to the XMCD standard.

Usage

getMobiusCapacities(tree, criteriaIDs, numberOfCriteria, kadditivity, mcdaConcept = NULL)

Arguments

tree Object containing the XMCD XML tree.
criteriaIDs A vector containing the IDs of the criteria to be considered for the extractions.
numberOfCriteria The total number of criteria to be considered.
kadditivity The level of k-additivity to be considered.
mcdaConcept A string containing the specific mcdaConcept attribute which should be searched for.

Value

The function returns a list structured as follows:

-- The first elements contain the <criteriaValues> found in <tree> as capacities (see kappaLab package for details). These elements are named according to the mcdaConcept attribute if it can be found.
status Either OK if all the <criteriaValues> tags could be correctly read, or the description of the error.

Examples

tree <- xmlTreeParse(system.file("extdata",
                       "testFile.xml",
                       package="RXMCDA"),
                       useInternalNodes=TRUE)

critIDs <- getCriteriaIDs(tree)

capa <- getMobiusCapacities(tree, critIDs[[1]], 5, 5,
                                  mcdaConcept="mobiusCapacity")
getDescription

Description

Extracts the number of alternatives from the <alternatives> tag from an XML tree written according to the XMCDA standard.

Usage

generateDescription(tree, mcdaConcept = NULL)

Arguments

tree Object containing the XMCDA XML tree.
mcdaConcept A string containing the specific mcdaConcept attribute which should be searched for.

Value

The function returns a list structured as follows:

-- The first elements contain the number of alternatives of each <alternatives> tag found in <tree>. These elements are named according to the mcdaConcept attribute if it can be found.
status Either OK if all the <alternatives> tags could be correctly read, or the description of the error.

Examples

tree = newXMLDoc()
	newXMLNode("xmcda:XMCDA",
	   namespace = c("xsi" = "http://www.w3.org/2001/XMLSchema-instance",
	   "xmcda" = "http://www.decision-deck.org/2009/XMCDA-2.0.0"),
	   parent=tree)

root<-getNodeSet(tree, "/xmcda:XMCDA")

alternatives<-newXMLNode("alternatives", parent=root[[1]], namespace=c())
	newXMLNode("alternative",attrs = c(id="x1"), parent=alternatives, namespace=c())
	newXMLNode("alternative",attrs = c(id="x2"), parent=alternatives, namespace=c())

y<-getNodeSet(tree,"//alternatives")

x<-getNumberOfAlternatives(y[[1]])
**getNumberOfCategories**  
*Get number of categories*

### Description

Extracts the number of categories from the `<categories>` tag from an XML tree written according to the XMCDA standard.

### Usage

```r
getNumberOfCategories(tree, mcdaConcept = NULL)
```

### Arguments

- **tree**: Object containing the XMCDA XML tree.
- **mcdaConcept**: A string containing the specific `mcdaConcept` attribute which should be searched for.

### Value

The function returns a list structured as follows:

- **--** The first elements contain the number of categories of each `<categories>` tag found in `<tree>`. These elements are named according to the `mcdaConcept` attribute if it can be found.
- **status**: Either OK if all the `<categories>` tags could be correctly read, or the description of the error.

### Examples

```r
tree = newXMLDoc()

newXmlNode("xmcda:XMCDA",
    namespace = c("xsi" = "http://www.w3.org/2001/XMLSchema-instance",
                  "xmcda" = "http://www.decision-deck.org/2009/XMCDA-2.0.0"),
    parent=tree)

root<-getNodeSet(tree, "/xmcda:XMCDA")
categories<-newXmlNode("categories", parent=root[[1]], namespace=c())

newXmlNode("category", attrs = c(id="c1"), parent=categories, namespace=c())
newXmlNode("category", attrs = c(id="c2"), parent=categories, namespace=c())

y<-getNodeSet(tree, "/categories")
x<-getNumberofCategories(y[[1]])
```
**getNumberOfCriteria**  
*Get number of criteria*

**Description**

Extracts the number of criteria from an XML tree written according to the XMCDA standard.

**Usage**

```
getNumberOfCriteria(tree, mcdaConcept = NULL)
```

**Arguments**

- **tree**: Object containing the XMCDA XML tree.
- **mcdaConcept**: A string containing the specific mcdaConcept attribute which should be searched for.

**Value**

The function returns a list structured as follows:

- **--**: The first elements contain the number of criteria of each `<criteria>` found in `<tree>`. These elements are named according to the mcdaConcept attribute if it can be found.
- **status**: Either OK if all the `<criteria>` tags could be correctly read, or the description of the error.

**Examples**

```r
tree = newXMLDoc()

newXmlNode("xmcda:XMCDA",
   namespace = c("xsi" = "http://www.w3.org/2001/XMLSchema-instance",
   "xmcda" = "http://www.decision-deck.org/2009/XMCDA-2.0.0"),
   parent=tree)

root<-getNodeSet(tree, "/xmcda:XMCDA")

criteria<-newXmlNode("criteria", parent=root[[1]], namespace=c())

criterion<-newXmlNode("criterion",attrs = c(id="g1"),
   parent=criteria, namespace=c())

y<-getNodeSet(tree,"/criteria")

x<-getNumberOfCriteria(y[[1]])
```
**getNumericValue**  
*Get a numeric value*

### Description

Extracts a numeric value (integer, real or rational) from an XML tree written according to the XMCDA standard and returns it as a real number.

### Usage

`getNumericValue(tree)`

### Arguments

- **tree**
  
  Object containing the XMCDA XML tree (in practice, only the part containing the `<value>` to be read).

### Value

The function returns a float based on the numeric value read.

### Examples

```plaintext
tree = newXMLDoc()
newXMLNode("xmcda:XMCDA",
    namespace = c("xsi" = "http://www.w3.org/2001/XMLSchema-instance",
    "xmcda" = "http://www.decision-deck.org/2009/XMCDA-2.0.0"),
    parent=tree)
root<-getNodeSet(tree, "/xmcda:XMCDA")
val<-newXMLNode("value", parent=root[[1]], namespace=c())
newXMLNode("real",3.14,parent=val, namespace=c())
y<-getNodeSet(tree,"//value")
x<-getNumericValue(y)
```
getParameters

Description

Gets the parameters defined in the <methodParameters> tag from an XML tree written according to the XMCDA standard.

Usage

getParameters(tree, name = NULL)

Arguments

tree Object containing the XMCDA XML tree.
name A string containing the specific name attribute which should be searched for.

Value

The function returns a list structured as follows:

- The first elements contain the parameters which have been read in each parameter tag (either a <label>, <real>, <integer> or <boolean>). These elements are named according to the name attribute if it can be found.

status Either OK if all the <methodParameters> tags could be correctly read, or the description of the error.

Examples

tree = newXMLDoc()

newXMLNode("xmcda:XMCDA",
    namespace = c("xsi" = "http://www.w3.org/2001/XMLSchema-instance",
    "xmcda" = "http://www.decision-deck.org/2009/XMCDA-2.0.0"),
    parent=tree)

root<-getNodeSet(tree, "/xmcda:XMCDA"

parameters<-newXMLNode("methodParameters", parent=root[[1]], namespace=c())

parameter <- newXMLNode("parameter", attrs = c(name="numIt"),
    parent=parameters, namespace=c())

value <- newXMLNode("value", parent=parameter, namespace=c())

newXMLNode("integer", value=3, parent=value, namespace=c())

y<-getNodeSet(tree,"//methodParameters"

x<-getParameters(y[[1]])
getPerformanceTables  Get performance tables

Description

Extracts the performance tables stored in the <performanceTable> tags from an XML tree written according to the XMCDA standard.

Usage

getPerformanceTables(tree, altIDs = NULL, critIDs = NULL, mcdaConcept = NULL)

Arguments

tree Object containing the XMCDA XML tree.
altIDs A vector containing the IDs of the alternatives to be considered for the extractions.
critIDs A vector containing the IDs of the criteria to be considered for the extractions.
mcdaConcept A string containing the specific mcdaConcept attribute which should be searched for.

Value

The function returns a list structured as follows:

- -- The first elements contain the performance tables found in each <performanceTable> of <tree> as matrices. The rows of each matrix are labelled according to the alternatives' IDs and the columns according to the criteria IDs. These elements are named according to the mcdaConcept attribute if it can be found.
status Either OK if all the <performanceTable> tags could be correctly read, or the description of the error.

Examples

tree <- xmlTreeParse(system.file("extdata","testFile.xml",package="RXMLCDA"),
                     useInternalNodes=TRUE)

tables <- getPerformanceTables(tree)
putAlternativesAffectations

Puts alternative affectations

Description

Puts alternative affectations as an `<alternativesAffectations>` tag in an XML tree written according to the XMCDA standard.

Usage

```r
putAlternativesAffectations(tree, alternativesAffectations, alternativesIDs, categoriesIDs, asIntervalsIfPossible = FALSE, mcdaConcept = NULL)
```

Arguments

- **tree**: Object containing the XMCDA XML tree.
- **alternativesAffectations**: An \( n \times m \) matrix representing `<alternativesAffectations>`, where \( n \) is number of considered alternatives and \( m \) is number of considered categories. Each cell \([i, j]\) of each matrix is boolean and corresponds to encoded relation between \( i \)-th alternative and \( j \)-th category.
- **alternativesIDs**: A vector containing the IDs of the alternatives to be considered for the extractions.
- **categoriesIDs**: A vector containing the IDs of the categories to be considered for the extractions.
- **asIntervalsIfPossible**: Whether put each affectation as a `<categoriesInterval>` tag (if possible) or as a `<categoriesSet>` tag.
- **mcdaConcept**: A string containing the specific `mcdaConcept` attribute which should be searched for.

Value

The function returns a list structured as follows:

- **status**: Either OK if all the `<alternativesAffectations>` tags could be correctly put, or the description of the error.

Examples

```r
alternativesIDs <- c("a01", "a02", "a03", "a04")
categoriesIDs <- c("c01", "c02", "c03", "c04")
altAff = rbind(c(FALSE, TRUE, TRUE, TRUE),
              c(FALSE, TRUE, FALSE, FALSE),
              c(TRUE, TRUE, TRUE, TRUE),
```
putAlternativesAffectationsWithValues

Puts alternative affectations with values

Description

Puts alternative affectations as an <alternativesAffectations> tag in an XML tree written according to the XMCDA standard.

Usage

putAlternativesAffectationsWithValues(tree, alternativesAffectations, alternativesIDs, categoriesIDs, mcdaConcept = NULL)

Arguments

tree Object containing the XMCDA XML tree.
alternativesAffectations An \( n \times 3 \) matrix representing <alternativesAffectations>, where \( n \) is the number of alternative affectations to write. Each row \([i, 1]\) of the matrix represents affectation of \([i, 1]\)-th alternative to \([i, 2]\)-th category with value \([i, 3]\).
alternativesIDs A vector containing the IDs of the alternatives to be considered for the extractions.
categoriesIDs A vector containing the IDs of the categories to be considered for the extractions.
mcdaConcept A string containing the specific mcdaConcept attribute which should be searched for.

Value

The function returns a list structured as follows:

status Either OK if all the <alternativesAffectations> tags could be correctly put, or the description of the error.
Examples

```r
alternativesIDs <- c("a01", "a02", "a03", "a04")
categoriesIDs <- c("c01", "c02", "c03")
altAff = rbind(c(1, 2, 0.1),
        c(1, 3, 0.0),
        c(2, 3, 1.0),
        c(4, 1, 0.5))

tree = newXMLDoc()

newXMLNode("xmcda:xMCDA",
    namespace = c("xsi" = "http://www.w3.org/2001/XMLSchema-instance",
                   "xmcda" = "http://www.decision-deck.org/2009/XMCDA-2.1.0"),
    parent=tree)

putAlternativesAffectationsWithValues(tree, altAff, alternativesIDs, categoriesIDs)
```

---

**putAlternativesComparisonsLabels**

*Put comparisons of alternatives*

**Description**

Writes `<alternativesComparisons>` in an XML tree written according to the XMCDA standard.

**Usage**

```r
putAlternativesComparisonsLabels(tree, alternativesComparisons,  
    mcdaConcept = NULL)
```

**Arguments**

- `tree`: Object containing the XMCDA XML tree.
- `alternativesComparisons`: A matrix containing the `<alternativesComparisons>` as a matrix. Each line of each matrix corresponds to one constraint of the type "alternative a is preferred to alternative b with preference threshold delta". A line is structured as follows: the first element encodes the ID of alternative a, the second element encodes the ID of alternative b, and the last element contains a valuation.
- `mcdaConcept`: A string containing the specific mcdaConcept attribute which should be written.

**Value**

The function returns a list structured as follows:

- `status`: Either OK if all the `<alternativesComparisons>` tags could be correctly put, or the description of the error.
**Examples**

```r
tree = newXMLDoc()

newXMLNode("xmcda:xmcda",
    namespace = c("xsi" = "http://www.w3.org/2001/XMLSchema-instance",
    "xmcda" = "http://www.decision-deck.org/2009/XMCDA-2.1.0"),
    parent=tree)

comps <- rbind(c("x", "y", "0.07"), c("y", "z", "0.01"))

altIDs <- c("x","y","z")

putAlternativesComparisonsLabels(tree, comps, mcdaConcept="newComparisons")
```

---

**putAlternativesIDs**  
*Put ids of alternatives*

**Description**

Puts ids of alternatives in an alternatives tag in an XML tree written according to the XMCDA standard.

**Usage**

```r
putAlternativesIDs(tree, alternativesIDs, mcdaConcept = NULL)
```

**Arguments**

- `tree`: Object containing the XMCDA XML tree.
- `alternativesIDs`: A vector containing the alternatives' IDs.
- `mcdaConcept`: A string containing the specific mcdaConcept attribute which should be written.

**Value**

The function returns a list structured as follows:

- `status`: Either OK if all the `<alternative>` tags could be correctly put, or the description of the error.

**Examples**

```r
altIDs <- c("x","y","z")

tree = newXMLDoc()

newXMLNode("xmcda:xmcda",
    namespace = c("xsi" = "http://www.w3.org/2001/XMLSchema-instance",
    "xmcda" = "http://www.decision-deck.org/2009/XMCDA-2.1.0"),
    parent=tree)
```
"xmcda" = "http://www.decision-deck.org/2009/XMCDA-2.1.0"),
parent=tree)

putAlternativesIDs(tree,altIDs,mcdaConcept="test")

altIDs2 <- getAlternativesIDs(tree)

putAlternativesPlot

Put plot in base64 related to alternatives

Description

Puts a plot coded in base64 as an alternativeValue tag in an XML tree written according to the XMCDA standard.

Usage

putAlternativesPlot(tree, base64plot, alternativesIDs,
                    mcdaConcept=NULL, name=NULL)

Arguments

tree          Object containing the XMCDA XML tree.
base64plot    String containing the base64 encoding of the plot.
alternativesIDs
mcdaConcept   A string containing the specific mcdaConcept attribute which should be written.
name           A string containing the specific name attribute which should be written.

Value

The function returns a list structured as follows:

status         Either OK if the <alternativeValue> tag could be correctly put, or the description of the error.
**putAlternativesValues**  
*Put values related to alternatives*

**Description**

Puts values related to alternatives as an `alternativesValues` tag in an XML tree written according to the XMCDA standard.

**Usage**

```r
putAlternativesValues(tree, alternativesValues, alternativesIDs, mcdaConcept = NULL)
```

**Arguments**

- `tree` Object containing the XMCDA XML tree.
- `alternativesValues` A matrix containing the values of the alternatives which have to be stored. Each line of this matrix represents a statement of the form "alternative x has value y". The first element of each line stores the index of the alternative x in `alternativesIDs`, the second element stores the value y.
- `alternativesIDs` A vector containing the alternatives' IDs.
- `mcdaConcept` A string containing the specific mcdaConcept attribute which should be written.

**Value**

The function returns a list structured as follows:

- `status` Either OK if all the `<alternativesValues>` tags could be correctly put, or the description of the error.

**Examples**

```r
altIDs <- c("x","y","z")
altVals <- rbind(c(1,1),c(2,0.5),c(3,0.2))

library(xmcda)

tree = newXMLDoc()

newXmlNode("xmcda:XMCDA",
  namespace = c("xsi" = "http://www.w3.org/2001/XMLSchema-instance",
  "xmcda" = "http://www.decision-deck.org/2009/XMCDA-2.1.0"),
  parent=tree)

putAlternativesValues(tree, altVals, altIDs)

altVals2 <- getAlternativesValues(tree, altIDs)
```

---

**Description**

Puts values related to alternatives as an `alternativesValues` tag in an XML tree written according to the XMCDA standard.

**Usage**

```r
putAlternativesValues(tree, alternativesValues, alternativesIDs, mcdaConcept = NULL)
```

**Arguments**

- `tree` Object containing the XMCDA XML tree.
- `alternativesValues` A matrix containing the values of the alternatives which have to be stored. Each line of this matrix represents a statement of the form "alternative x has value y". The first element of each line stores the index of the alternative x in `alternativesIDs`, the second element stores the value y.
- `alternativesIDs` A vector containing the alternatives' IDs.
- `mcdaConcept` A string containing the specific mcdaConcept attribute which should be written.

**Value**

The function returns a list structured as follows:

- `status` Either OK if all the `<alternativesValues>` tags could be correctly put, or the description of the error.

**Examples**

```r
altIDs <- c("x","y","z")
altVals <- rbind(c(1,1),c(2,0.5),c(3,0.2))

library(xmcda)

tree = newXMLDoc()

newXmlNode("xmcda:XMCDA",
  namespace = c("xsi" = "http://www.w3.org/2001/XMLSchema-instance",
  "xmcda" = "http://www.decision-deck.org/2009/XMCDA-2.1.0"),
  parent=tree)

putAlternativesValues(tree, altVals, altIDs)

altVals2 <- getAlternativesValues(tree, altIDs)
```
putAlternativeValue  Put a value related to an alternative

Description

Puts a value related to an alternative (or a set of alternatives) as a alternativeValue tag in an XML tree written according to the XMCDA standard.

Usage

putAlternativeValue(tree, alternativeValue, alternativesIDs = NULL, mcdaConcept = NULL)

Arguments

tree  Object containing the XMCDA XML tree.

alternativeValue  A float representing the value assigned to the alternative.

alternativesIDs  A string representing the alternative’s ID or a vector representing a set of alternatives IDs.

mcdaConcept  A string containing the specific mcdaConcept attribute which should be written.

Value

The function returns a list structured as follows:

status  Either OK if the <alternativeValue> tag could be correctly put, or the description of the error.

Examples

altID <- c("x")

tree = newXMLDoc()

newXMLNode("xmcda:XMCDA",
    namespace = c("xsi" = "http://www.w3.org/2001/XMLSchema-instance",
    "xmcda" = "http://www.decision-deck.org/2009/XMCDA-2.1.0"),
    parent=tree)

putAlternativeValue(tree, 0.8, alternativesIDs = altID)
**Description**

Puts values related to a capacity as a criteriaValues tag in an XML tree written according to the XMCDA standard.

**Usage**

```r
putCapacity(tree, capacity, criteriaIDs, mcdaConcept = NULL)
```

**Arguments**

- `tree` Object containing the XMCDA XML tree.
- `capacity` An object of type capacity (see kappa lab package).
- `criteriaIDs` A vector containing the criteria’s IDs.
- `mcdaConcept` A string containing the specific mcdaConcept attribute which should be written.

**Value**

The function returns a list structured as follows:

- `status` Either OK if all the <criteriaValues> tags could be correctly put, or the description of the error.

**Examples**

```r
library(kappalab)

tree = newXMLDoc()

newXMLNode("xmcda:XMCDA",
    namespace = c("xsi" = "http://www.w3.org/2001/XMLSchema-instance",
    "xmcda" = "http://www.decision-deck.org/2009/XMCDA-2.1.0"),
    parent=tree)

mu<-capacity(0:15)
a <- Mobius(mu)

critIDs <- c("g1","g2","g3","g4")

putCapacity(tree, a, critIDs, mcdaConcept="capacity")
```
putCategoriesIntervalValues

Put interval values related to categories

Description

Puts interval values related to categories as a categoriesValues tag in an XML tree written according to the XMCDA standard.

Usage

putCategoriesIntervalValues(tree, categoriesValues, categoriesIDs, mcdaConcept = NULL)

Arguments

tree Object containing the XMCDA XML tree.
categoriesValues A matrix representing a <categoriesValues>. Each line of each matrix corresponds to an element of the type "the interval value assigned to category c1 is [x, y]". A line is structured as follows: the first element encodes the index of the ID of category c1 in categoriesIDs, the second element encodes the value x and the third element encodes the value y. These elements are named according to the mcdaConcept attribute if it can be found.
categoriesIDs A vector containing the categories’ IDs.
mcdaConcept A string containing the specific mcdaConcept attribute which should be written.

Value

The function returns a list structured as follows:

status Either OK if all the <categoriesValues> tags could be correctly put, or the description of the error.

Examples

categoriesIDs <- c("c01", "c02", "c03", "c04")
categoriesIntervalValues <- rbind(c(1, 0.4, 0.7), c(2, 0.5, 0.5), c(4, 0.2, 0.9))

tree = newXMLDoc()

newXMLNode("xmcda:XMCDA",
    namespace = c("xsi" = "http://www.w3.org/2001/XMLSchema-instance",
                  "xmcda" = "http://www.decision-deck.org/2009/XMCDA-2.1.0"),
    parent=tree)

putCategoriesIntervalValues(tree, categoriesIntervalValues, categoriesIDs)
**putCategoriesValues**  
*Put values related to categories*

**Description**

Puts values related to categories as a `categoriesValues` tag in an XML tree written according to the XMCDA standard.

**Usage**

```r
putCategoriesValues(tree, categoriesValues, categoriesIDs, 
                   mcdaConcept = NULL)
```

**Arguments**

- **tree**: Object containing the XMCDA XML tree.
- **categoriesValues**: A matrix representing a `<categoriesValues>` tag. Each line of each matrix corresponds to an element of the type "the value(s) assigned to category c1 is (are) x (x1, x2, ...)". A line is structured as follows: the first element encodes the index of the ID of category c1 in `categoriesIDs`, next elements encode the values x1, x2... These elements are named according to the `mcdaConcept` attribute if it can be found.
- **categoriesIDs**: A vector containing the categories’ IDs.
- **mcdaConcept**: A string containing the specific `mcdaConcept` attribute which should be written.

**Value**

The function returns a list structured as follows:

- **status**: Either OK if all the `<categoriesValues>` tags could be correctly put, or the description of the error.

**Examples**

```r
categoriesIDs <- c("c01", "c02", "c03", "c04")
categoriesValues <- rbind(c(1, 0.4), c(2, 0.5), c(4, 0.2))

# Create an XML tree

# Create XML node

# Put categories values

putCategoriesValues(tree, categoriesValues, categoriesIDs)
```
putCriteriaMatrix  Puts values in a criteria matrix

Description

Puts values related to a matrix of criteria as a criteriaMatrix tag in an XML tree written according to the XMCDA standard.

Usage

putCriteriaMatrix(tree, criteriaMatrix, mcdaConcept = NULL)

Arguments

tree  Object containing the XMCDA XML tree.
criteriaMatrix  A matrix representing a <criteriamatrix>. The lines and the columns are named according to criteria IDs.
mcdaConcept  A string containing the specific mcdaConcept attribute which should be written.

Value

The function returns a list structured as follows:

status  Either OK if the <criteriamatrix> tag could be correctly put, or the description of the error.

Examples

critIDs <- c("x","y","z")
criteriaMatrix <- rbind(c(1,2,3),c(4,5,6),c(7,8,9))
rownames(criteriaMatrix) <- critIDs
colnames(criteriaMatrix) <- critIDs
tree = newXMLDoc()
newXMLNode("xmcda:XMCDA",
  namespace = c("xsi" = "http://www.w3.org/2001/XMLSchema-instance",
  "xmcda" = "http://www.decision-deck.org/2009/XMCDA-2.1.0"),
  parent=tree)
putCriteriaMatrix(tree, criteriaMatrix)
**putCriteriaPairsValues**

*Put values related to pairs of criteria*

**Description**

Puts values related to pairs of criteria as a criteriaValues tag in an XML tree written according to the XMCDA standard.

**Usage**

```
putCriteriaPairsValues(tree, criteriaPairsValues, criteriaIDs,
                       mcdaConcept = NULL)
```

**Arguments**

- **tree**: Object containing the XMCDA XML tree.
- **criteriaPairsValues**: A as matrix representing the values assigned to pairs of criteria. Each line of the matrix corresponds to one statement of the type "the value assigned to the couple of criteria (g1,g2) is x". A line is structured as follows: the first element encodes the index of criterion g1 in criteriaIDs, the next element encodes the index of the ID of criterion g2, and the last elements contain x.
- **criteriaIDs**: A vector containing the criteria’s IDs.
- **mcdaConcept**: A string containing the specific mcdaConcept attribute which should be written.

**Value**

The function returns a list structured as follows:

- **status**: Either OK if all the <criteriaValues> tags could be correctly put, or the description of the error.

**Examples**

```
tree = newXMLDoc()

newXMLNode("xmcda:XMCDA",
           namespace = c("xsi" = "http://www.w3.org/2001/XMLSchema-instance",
                          "xmcda" = "http://www.decision-deck.org/2009/XMCDA-2.1.0"),
                          parent=tree)

critIDs <- c("g1","g2","g3","g4")

pairsVals <- rbind(c(1,2,0.17),c(2,3,0.5), c(3,4,0.16))

putCriteriaPairsValues(tree,pairsVals,critIDs)
```
putCriteriaPlot  
*Put a plot in base64 related to criteria*

**Description**

Puts a plot coded in base64 as a criterionValue tag in an XML tree written according to the XMCDA standard.

**Usage**

```r
putCriteriaPlot(tree, base64plot, criteriaIDs, mcdaConcept=NULL, name=NULL)
```

**Arguments**

- `tree`  
  Object containing the XMCDA XML tree.
- `base64plot`  
  String containing the base64 encoding of the plot.
- `criteriaIDs`  
  A vector of criteria’s IDs.
- `mcdaConcept`  
  A string containing the specific mcdaConcept attribute which should be written.
- `name`  
  A string containing the specific name attribute which should be written.

**Value**

The function returns a list structured as follows:

- `status`  
  Either OK if the `<criterionValue>` tag could be correctly put, or the description of the error.

---

putCriteriaValues  
*Put values related to criteria*

**Description**

Puts values related to criteria as a criteriaValues tag in an XML tree written according to the XMCDA standard.

**Usage**

```r
putCriteriaValues(tree, criteriaValues, criteriaIDs, 
                  mcdaConcept = NULL)
```

**Value**

The function returns a list structured as follows:

- `status`  
  Either OK if the `<criteriaValues>` tag could be correctly put, or the description of the error.
putCriterionValue

Arguments

- **tree**: Object containing the XMCDA XML tree.
- **criteriValues**: A matrix representing a `<criteriValues>` tag. Each line of the matrix corresponds to an element of the type "the value assigned to criterion g1 is x". A line is structured as follows: the first element encodes the index of the ID of criterion g1 in `criteriaIDs`, the second element encodes the value x. These elements are named according to the mcdaConcept attribute if it can be found.
- **criteriaIDs**: A vector containing the criteria’s IDs.
- **mcdaConcept**: A string containing the specific mcdaConcept attribute which should be written.

Value

The function returns a list structured as follows:

- **status**: Either OK if all the `<criteriValues>` tags could be correctly put, or the description of the error.

Examples

```r
critIDs <- c("x","y","z")
critVals <- rbind(c(1,1),c(2,0.5),c(3,0.2))
tree = newXMLDoc()

newXMLNode("xmcda:XMCDA",
    namespace = c("xsi" = "http://www.w3.org/2001/XMLSchema-instance",
    "xmcda" = "http://www.decision-deck.org/2009/XMCDA-2.1.0"),
    parent=tree)

putCriteriaValues(tree, critVals, critIDs)
```

---

**putCriterionValue**: Put a value related to a criterion

Description

Puts a value related to a criterion (or a set of criteria) as a criterionValue tag in an XML tree written according to the XMCDA standard.

Usage

```
putCriterionValue(tree, criterionValue, criteriaIDs = NULL,
    mcdaConcept = NULL)
```
Arguments

tree Object containing the XMCDA XML tree.

criterionValue A float representing the value assigned to the criterion.
criteriaIDs A string representing the criterion’s ID or a vector representing a set of criteria IDs.
mcdaConcept A string containing the specific mcdaConcept attribute which should be written.

Value

The function returns a list structured as follows:

status Either OK if all the <criterionValue> tags could be correctly put, or the description of the error.

Examples

critID <- c("x")
tree = newXMLDoc()

newXMLNode("xmcda:XMCDA",
    namespace = c("xsi" = "http://www.w3.org/2001/XMLSchema-instance",
    "xmcda" = "http://www.decision-deck.org/2009/XMCDA-2.1.0"),
    parent=tree)

putCriterionValue(tree, 0.8, criteriaIDs = critID)

putError_Message

Put an error message

Description

Puts an error message in a <methodMessages> tag of an XML tree written according to the XMCDA standard.

Usage

putError_Message(tree, errorMessage, name = NULL)

Arguments

tree Object containing the XMCDA XML tree.
errorMessage A string containing the text of the error message.
name A string containing the specific name attribute which should be searched for.
**putLogMessage**

Value

The function returns a list structured as follows:

| status | Either OK if the `<errorMessage>` tag could be correctly put, or the description of the error. |

---

**Description**

Puts an log message in a `<methodMessages>` tag of an XML tree written according to the XMCDA standard.

**Usage**

```plaintext
putLogMessage(tree, logMessage, name = NULL)
```

**Arguments**

- **tree**: Object containing the XMCDA XML tree.
- **logMessage**: A string containing the text of the log message.
- **name**: A string containing the specific `name` attribute which should be searched for.

Value

The function returns a list structured as follows:

| status | Either OK if the `<logMessage>` tag could be correctly put, or the description of the error. |

---

**putMessage**

**Put a message**

**Description**

Puts an message in a `<methodMessages>` tag of an XML tree written according to the XMCDA standard.

**Usage**

```plaintext
putMessage(tree, message, name = NULL)
```
**putPerformanceTable**

**Arguments**

- **tree**: Object containing the XMCDA XML tree.
- **message**: A string containing the text of the log message.
- **name**: A string containing the specific name attribute which should be put.

**Value**

The function returns a list structured as follows:

- **status**: Either OK if the `<message>` tag could be correctly put, or the description of the error.

**Description**

Puts a performance table (performancetable tag) in an XML tree written according to the XMCDA standard.

**Usage**

```lisp
putPerformanceTable(tree, performancetable, mcdaConcept = NULL)
```

**Arguments**

- **tree**: Object containing the XMCDA XML tree.
- **performancetable**: A matrix representing the performance table (the lines are named according to the alternatives' IDs, the columns are named according to the criteria IDs).
- **mcdaConcept**: A string containing the specific mcdaConcept attribute which should be written.

**Value**

The function returns a list structured as follows:

- **status**: Either OK if the `<performancetable>` tag could be correctly put, or the description of the error.
Examples

```r
performanceTable <- rbind(c(1,2,3),c(4,5,6))
rownames(performanceTable) <- c("x","y")
colnames(performanceTable) <- c("g1","g2","g3")
tree = newXMLDoc()
	newXMLNode("xmcda:xmcda",
.namespace = c("xsi" = "http://www.w3.org/2001/XMLSchema-instance",
"xmcda" = "http://www.decision-deck.org/2009/XMCD//2.1.0"),
.parent=tree)

putPerformanceTable(tree,performanceTable)
```

Description

Put value functions defined by sets of points in a criterionFunction tag under the criterion tag in an XML tree written according to the XMCDA standard.

Usage

```r
putPointsCriterionFunction(tree, points, mcdaConcept = NULL)
```

Arguments

- **tree**: Object containing the XMCDA XML tree.
- **points**: A list, where each element is named by the ID of a criterion, and contains a matrix representing the points (each line is a point, the first column represents the abscissa, the second the ordinate).
- **mcdaConcept**: A string containing the specific mcdaConcept attribute which should be written.

Value

The function returns a list structured as follows:

- **status**: Either OK if all the criterionFunction> tags could be correctly put, or the description of the error.
Examples

tree = newXMLDoc()

newXMLNode("xmcda:XMCDA",
    namespace = c("xsi" = "http://www.w3.org/2001/XMLSchema-instance",
    "xmcda" = "http://www.decision-deck.org/2009/XMCDA-2.1.0"),
    parent=tree)

x<-list()
x<-c(x,list(g1=cbind(c(1,2),c(3,4))))
x<-c(x,list(g2=cbind(c(5,6),c(7,8),c(9,10))))
x<-c(x,list(g3=cbind(c(11,12))))
x<-c(x,list(g4=cbind(c(13,14),c(15,16))))

putPointsCriterionFunction(tree,x)
Index

**Topic methods**
- checkxsd, 2
- getAlternativesAffectations, 3
- getAlternativesComparisons, 4
- getAlternativesComparisonsLabels, 6
- getAlternativesIDs, 8
- getAlternativesIntervalValues, 10
- getAlternativesValues, 11
- getCategoriesIDs, 12
- getCategoriesIntervalValues, 13
- getCategoriesValues, 14
- getCriteriaComparisons, 15
- getCriteriaComparisonsLabels, 16
- getCriteriaIDs, 17
- getCriteriaIntervalValues, 18
- getCriteriaPairsComparisons, 19
- getCriteriaPairsIntervalValues, 20
- getCriteriaPairsValues, 21
- getCriteriaValues, 22
- getMobiusCapacities, 23
- getNumberOfAlternatives, 24
- getNumberOfCategories, 25
- getNumberOfCriteria, 26
- getNumericValue, 27
- getParameters, 28
- getPerformanceTables, 29
- putAlternativesAffectations, 30
- putAlternativesAffectationsWithValues, 31
- putAlternativesComparisonsLabels, 32
- putAlternativesIDs, 33
- putAlternativesPlot, 34
- putAlternativesValues, 35
- putAlternativeValue, 36
- putCapacity, 37
- putCategoriesIntervalValues, 38
- putCategoriesValues, 39
- putCriteriaMatrix, 40
- putCriteriaPairsValues, 41
- putCriteriaPlot, 42
- putCriteriaValues, 42
- putCriterionValue, 43
- putErrorMessage, 44
- putLogMessage, 45
- putMessage, 45
- putPerformanceTable, 46
- putPointsCriterionFunction, 47

checkxsd, 2
- getAlternativesAffectations, 3
- getAlternativesComparisons, 4
- getAlternativesComparisonsLabels, 6
- get AlternativesComparisonsValues, 7
- getAlternativesIDs, 8
- getAlternativesIntervalValues, 10
- getAlternativesValues, 11
- getCategoriesIDs, 12
- getCategoriesIntervalValues, 13
- getCategoriesValues, 14
- getCriteriaComparisons, 15
- getCriteriaComparisonsLabels, 16
- getCriteriaIDs, 17
- getCriteriaIntervalValues, 18
- getCriteriaPairsComparisons, 19
- getCriteriaPairsIntervalValues, 20
- getCriteriaPairsValues, 21
- getCriteriaValues, 22
- getMobiusCapacities, 23
- getNumberOfAlternatives, 24
- getNumberOfCategories, 25
- getNumberOfCriteria, 26
- getNumericValue, 27
- getParameters, 28
- getPerformanceTables, 29

putAlternativesAffectations, 30
- putAlternativesPlot, 34
- putAlternativesValues, 35
- putAlternativeValue, 36
- putCapacity, 37
- putCategoriesIntervalValues, 38
- putCategoriesValues, 39

putAlternativesAffectations, 30
putAlternativesAffectationsWithValues, 31
putAlternativesComparisonsLabels, 32
putAlternativesIDs, 33
putAlternativesPlot, 34
putAlternativesValues, 35
putAlternativeValue, 36
putCapacity, 37
putCategoriesIntervalValues, 38
putCategoriesValues, 39
putCriteriaMatrix, 40
putCriteriaPairsValues, 41
putCriteriaPlot, 42
putCriteriaValues, 42
putCriterionValue, 43
putErrorMessage, 44
putLogMessage, 45
putMessage, 45
putPerformanceTable, 46
putPointsCriterionFunction, 47