ApacheCon Europe
Sinsheim, Germany 5th–8th November 2012

Change Tracking in AOO and ODF

Oliver-Rainer Wittmann
IBM

Svante Schubert
Freelancer

Peter Rakyta
MULTIRÁCIÓ
multitalent solutions

Supported by: GOP-1.1.1-11-2011-0006

Agenda

- Introduction
  - The people
  - The topic
- Change Tracking (CT) in AOO
  - Focus on word processor (Writer)
- CT in ODF
  - CT in ODF 1.2
  - Merge-enabled CT (MCT) proposal
- Extension prototype

Change Tracking in AOO – 1

- What is working in AOO Writer
  - Insertion and deletion of text
  - Basic text formatting
    (only tracked – not possible to reject)
  - Insertion of empty table or section
  - Deletion of table or section inclusive content
  - Insertion and deletion of a comment, a field, a footnote or an endnote
  - Insertion and deletion of text inside a text frame, a page header or a page footer

Change Tracking in AOO – 2

- What is not working in AOO Writer
  - Creation of a table from existing content
  - Deletion of a table without removing its content
  - Insertion and deletion of a table row/column
  - Merge and split of table cells
  - Insertion of a section around existing content
  - Deletion of a section without removing its content
  - Creation of a list or a list item from existing content
  - Deletion of a list or a list item without removing its content

Change Tracking in AOO – 3

- What is not working in AOO Writer – cont.
  - Correct overlapping and nesting of changes
  - Insertion and deletion of an anchored object (text frame, graphic, embedded object, drawing object)
  - Insertion and deletion of text inside a comment
  - Insertion and deletion of a page header, a page footer or a bookmark
  - Full support for tracking of format changes
  - Changes to attributes and styles
  - Grouping of changes – e.g. document-wide replace

Change Tracking in AOO – 4

- Obviously, the CT feature in Writer needs “attention”
  - Interoperability with Microsoft Word's CT has also been identified as an area of improvement [1]
- Reasons for the insufficiency in Writer:
  - Missing, incomplete and incorrect implementation in AOO Writer
  - Missing capabilities/support in the change tracking feature of ODF [2]

[1] http://markmail.org/message/g4a7ndokhf2ztmxw
[2] ODF versions 1.0, 1.1 and 1.2

Change Tracking in ODF 1.2

- Basic concepts of “Insertion”, “Deletion” and “Format change” for text documents
  - “Format change” is incomplete
  - No support for changes to attributes and styles
  - Inadequacy of ODF specification, especially regarding “Deletion” - see issues linked in [1]
  - No support for nesting and grouping
- Result of CT discussion at OASIS:
  - Creation of Collab SC
  - Three CT proposals – 'GCT', 'ECT' and 'MCT'

[1] https://tools.oasis-open.org/issues/browse/OFFICE-3312

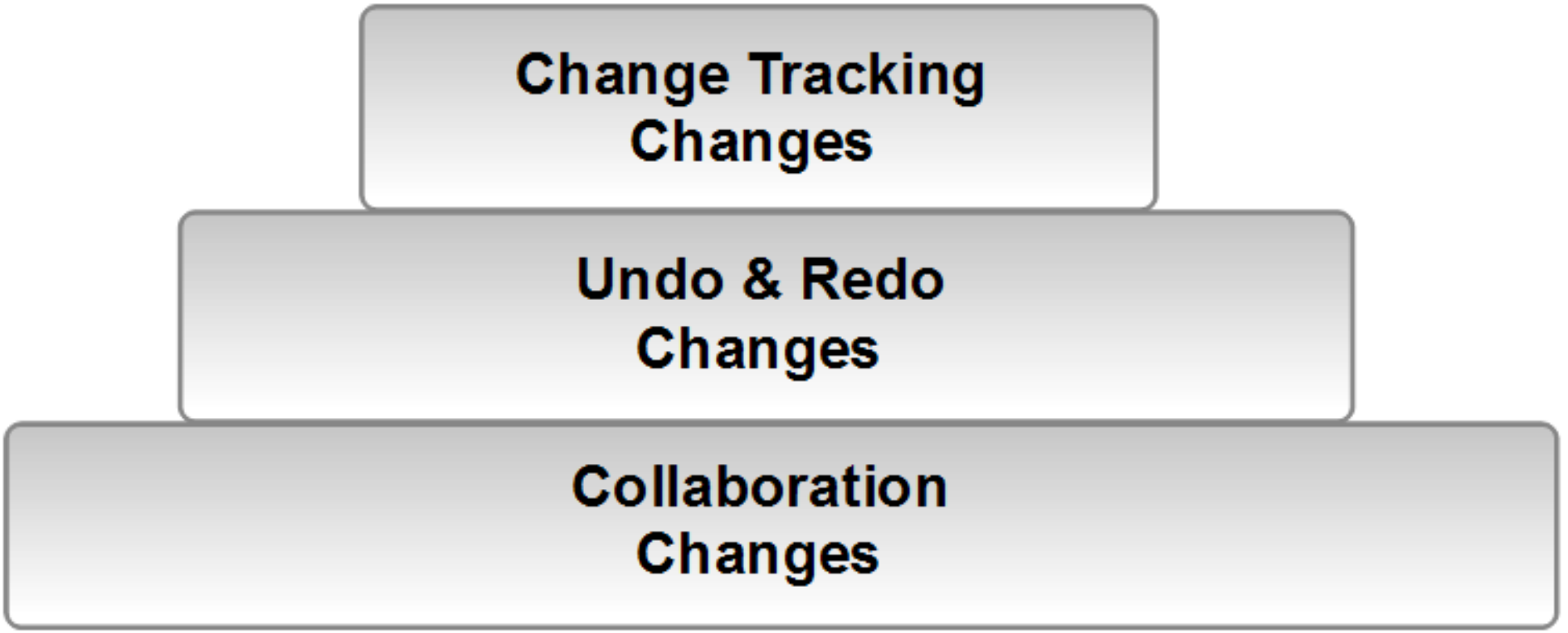
Merge-enabled CT proposal for next version of ODF

Svante Schubert
Freelancer

Change-Tracking for Dummies

- Basic change-tracking Problem
  - No standard / documentation what is changeable
  - XML grammar only describes the allowed XML documents
  - CT by before and after XML in file
  - No other file changes are being tracked (e.g. no GIT-like support)

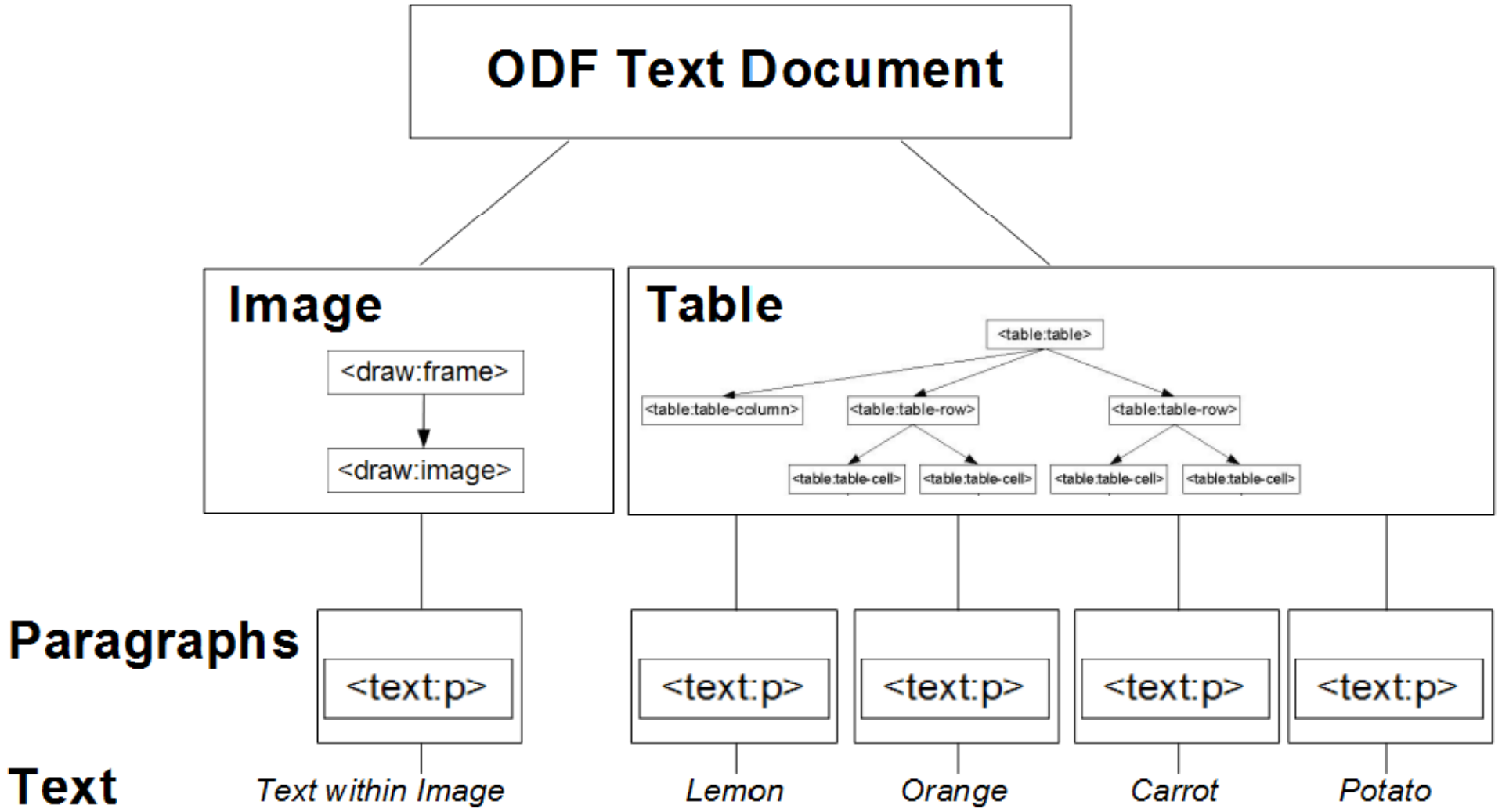
Dependencies of Changes



Basic MCT Design – Changes by Operation Calls

- Changes as in Real-Time Collaboration
- Convention over Configuration Pattern
  - Move repeating patterns into spec
  - Operation = XML change pattern
  - Label of change defined in spec (e.g. moveRow)
  - Conventions on empty document, default styles palettes, etc.

Semantic groups of ODF XML (Components)



Basic MCT Design – Basic Operations

- The four Operation Types:
  - **Add**
  - **Delete**
  - **Move** (special case of Add & Delete needed for efficiency)
  - **Replace** (special case of Add & Delete needed for efficiency)

Serializing Operations

Starting markup:

```
<text:h>Some boring, but important text!</text:h>
```

Ending markup:

```
<text:h>Some important text!</text:h>
```

Do changes:

```
<do>
   <del s=”/1/5” e=”/1/16” />
</do>
```

Undo changes:

```
<undo>
   <add s="/1/5"> boring, but</add>
</undo>
```

Serializing Example: Adding/Removing a List Level

Starting markup:

```
<text:list>
  <text:list-item><text:p>Line 1</text:p></text:list-item>
  <text:list-item><text:p>Line 2</text:p></text:list-item>
  <text:list-item><text:p>Line 3</text:p></text:list-item>
</text:list>
```

Ending markup:

```
<text:p>Line 1</text:p>
<text:p>Line 2</text:p>
<text:p>Line 3</text:p>
```

Changes (only FYI - not being saved):

```
<do>
   <del type=”list-level” s="/1" e=”/3”>
 </do>
```

Undo changes (undo.xml):

```
<undo>
   <add type=”unordered-list” s="/1" e="/3" />
</undo>
```

The Operation Queue/Stack

Status - 1

```
<do>
   <add type=”paragraph” s="/5">My work!</add>
</do>
```

Status - 2

```
<do>
   <add type=”paragraph” s="/2">Colleagues work!</add>
   <add type=”paragraph” s="/5">My work!</add>
</do>
```

Status - 2

```
<do>
   <add type=”paragraph” s="/2">Colleagues work!</add>
   <add type=”paragraph” s="/5">My work!</add>
</do>
```

Status - 3

```
<do>
   <add type=”paragraph” s="/6">My work!</add>
   <add type=”paragraph” s="/2">Colleagues work!</add>
</do>
```

Time
(latest on top)

Moving
operation
through time

Basic Operation Rules

- On top the newest (no dependencies) able to delete
- On bottom the oldest (all dependencies) able to apply
- Compression – multiple ops as one (e.g. characters)
- Normalization – most compressed and ordered as doc

Merging Changes

Start: User A

<add type=”paragraph” s="/5">My work!</add>

Start:User B

<add type=”paragraph” s="/2">Colleagues work!</add>

User A syncs with B

- Pull changes from B
- Check if changes from B influence own changes
  (Put ops stack of B on top of own and move every B op to bottom of A stack)
- After all ops of B passed (& influenced changes of A), push adapted A stack to B
- B could now simply put the (adapted) A stack on top of B

End: User A+B

<add type=”paragraph” s="/6">My work!</add>
<add type=”paragraph” s="/2">Colleagues work!</add>

Basic MCT Design - Relative reference to changes

- Advantage of changes outside ODF XML
  - Allow to apply changes on a read-only document (signed/in the web)
  - Merging changes of huge documents does not require the doc
  - Merging changes independent of document size

Low hanging Juicy Fruits

- Collaboration across ODF applications
- Merge functionality
- Arbitrary change grouping
  and moving changes through time
- Document read-only content (signed/web)
- History functionality

Extension prototype implementing MCT

Peter Rakyta

Supported by: GOP-1.1.1-11-2011-0006

Why developing it as an extension?

- The code of the extension is separated from the code lines of the OpenOffice.
- Easier development while the specification of MCT is still under construction.
- The same functionality can be adopted in various OpenOffice based applications.

The main structure of the extension

1. The model

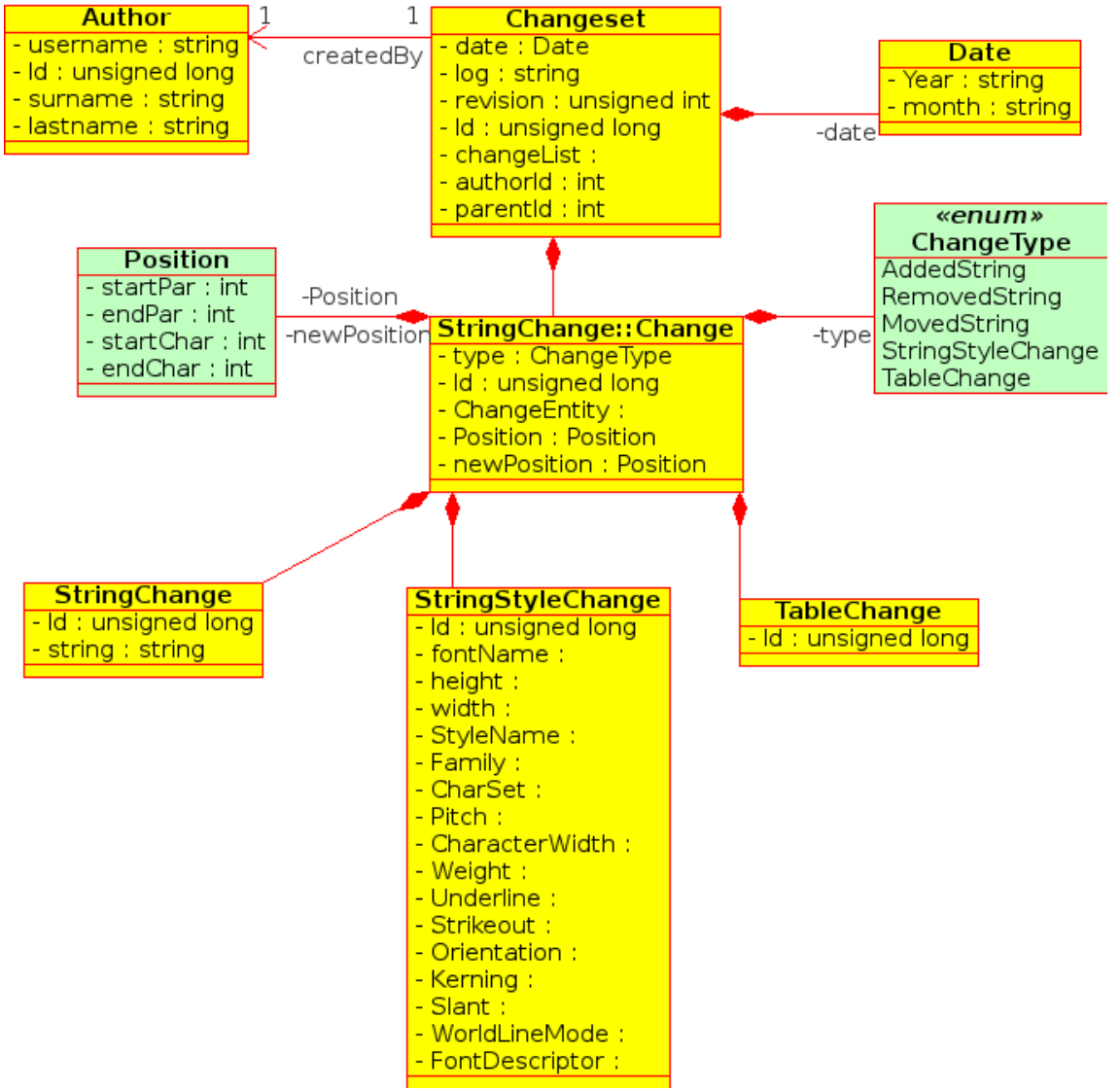
- Changes stored in a redo/undo graph.
- Should obey to the same specification in all implementation.
- XML parser to load and save the history of changes.
- Changes are grouped under changesets (revisions of the document).
- Storing other metadata for the changesets (author, date, comment).

2. Connection of the model to the ODF

- Established by the UNO Accessibility API.
- Change tracking based on event listener interfaces provided by the UNO.
- Dialog window in order to control the functionality of the extension.

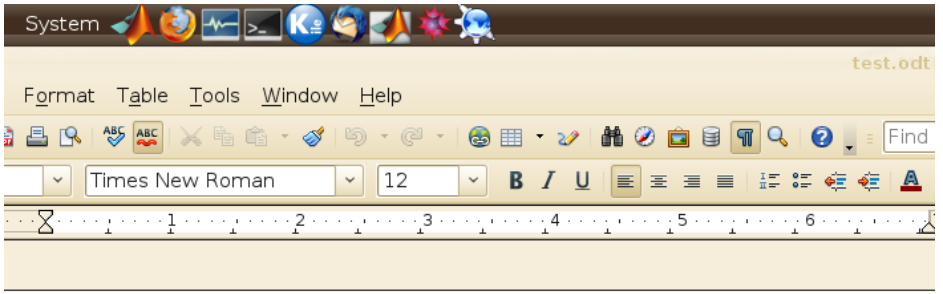
The Graph of change history

- In the current version string changes are supported
- Attributes of other “change nodes” need to be further specified.



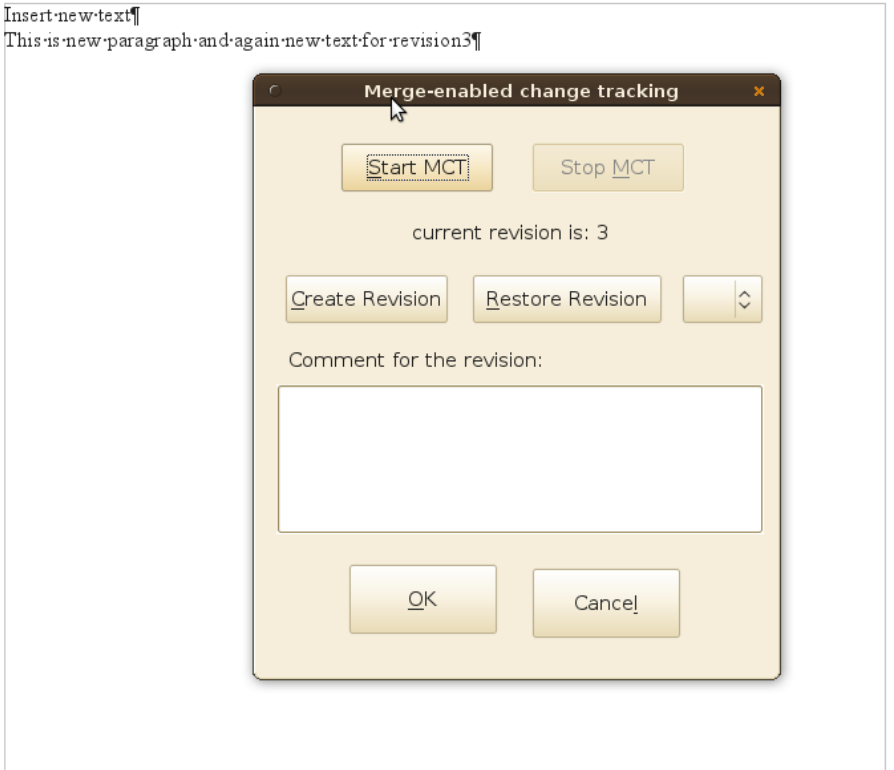
The dialog window

- **Create Revision:** collect changes and make a changeset in the graph. Exports change history into undo.xml.



- **Restore Revision**: restores the document into an earlier state. Fills redo graph with the data of the undid changes. Exports undo/redo graph into undo.xml/redo.xml.

- **Start/Stop MCT:** starting/stopping MCT support for the document.



Further Developments

- Merging two graphs (editions of two authors in the same document).
  - Accepting/rejecting changesets (dialog window)
  - Rules for merging overlapping changes
- Expand the range of supported change types.
  - StringStyleChange
  - TableChange
  - FigureChange
  - ...
- Capabilities to manage other metadata (storing data on the deleted figures to be able fully restore undid changes)
- Change Tracking in spreadsheets. (needs to re-design some nodes in the graph)