LibreOffice
The Document Foundation

Interoperable Office Collaboration

Svante Schubert

Svante.Schubert@gmail.com

TIRANA | 26 Sept. 2018

LIBOCON
TIRANA 2018

Question:

Will **Libreoffice in 50 years**

be still **our favorite editor**?

Answer:

Depends if **LibreOffice** will support the **state of the art features** in 50 years!

Bold Claim:

“Collaboration”
the most critical feature
in our connected world

Everyone of us has at two computer at least! -- laptop and smartphone!

Feature:

Collaborative real-time editor (2 modes)

1) **Real-Time Mode** (e.g. Etherpad, Google Docs, etc.)
Users can edit the same document simultaneously.

2) **Non-Real-Time Mode** (similar revision control systems)
Users edit a copy of document and merge later.

https://en.wikipedia.org/wiki/Collaborative_real-time_editor

Feature:

Collaborative real-time editor (2 modes)

In the end all “copies” are the same!

1) **Real-Time Mode**

   Automatic fix of merge conflicts! (for convenience).

2) **Non-Real-Time Mode**

   Merge conflicts have to be resolved by the user!

Merge conflict like I am editing a cell, YOU delete the table!

Feature:

Collaborative real-time editor (2 modes)

1) **Real-Time Mode**

Good for working with a group of trusted members.

2) **Non-Real-Time Mode**

Users like to be in control of all changes.

Legal departments of two companies collaborating.

Feature:

Collaborative real-time editor (2 modes)

1) **Real-Time Mode**

    Comes first to mind!

2) **Non-Real-Time Mode**

    Often forgotten! But IMHO most money is here!

How do real-time editors work?

- No documents are dispatched!

  Dispatching documents is stupid!!!

  As stupid as developers sending software repos!

- Sending changes / operations / differences / DIFFs!

  Best not text/syntax based, but **semantic changes**!

Today‘s Problem

The most important question in collaboration:
“What have you changed in the doc?”

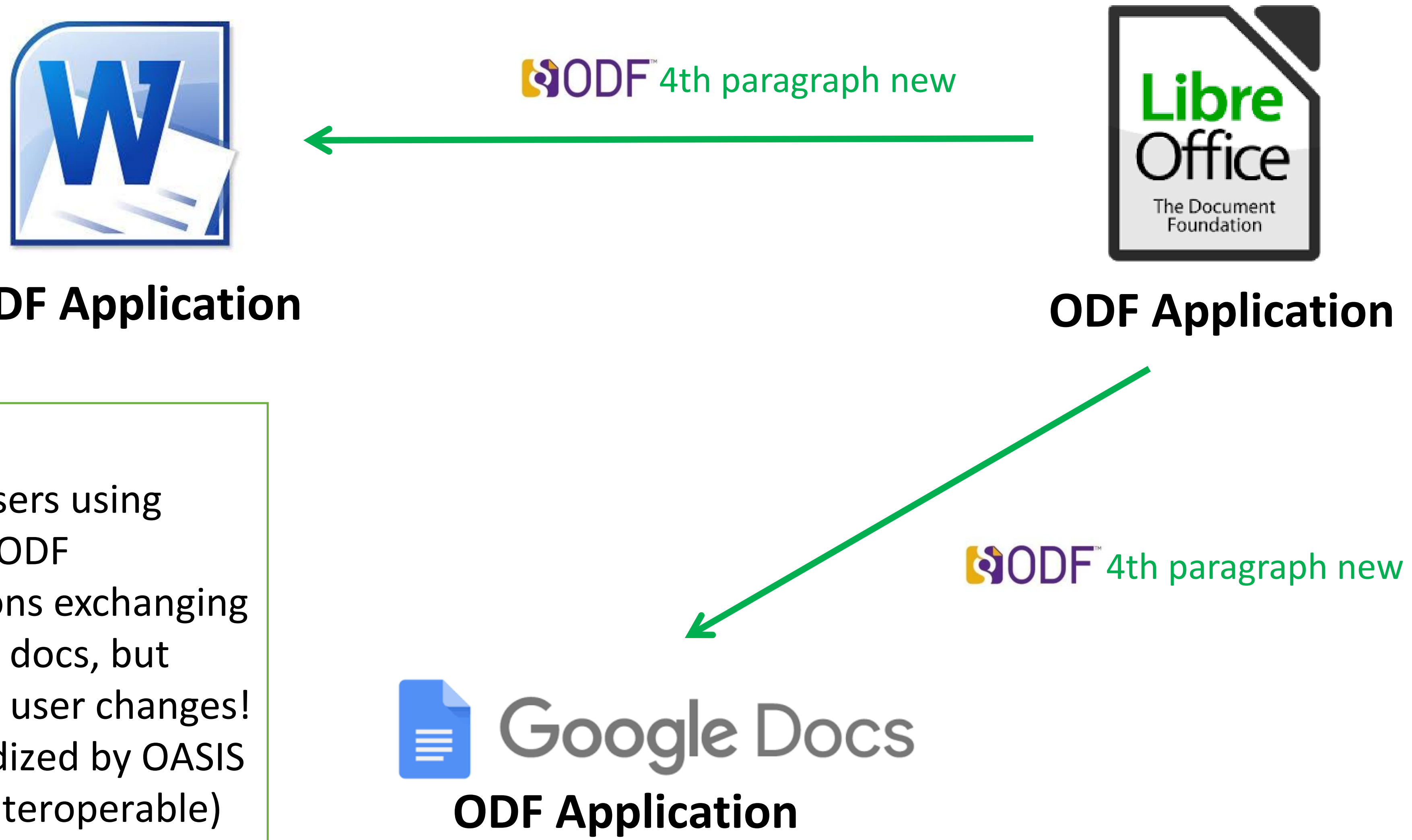
No way to answer in an **interoperable** way!
Only standards for file formats exist!
No standard for file changes!

How the Future might look like…

LIBOCON TIRANA 2018

Interoperable Collaboration

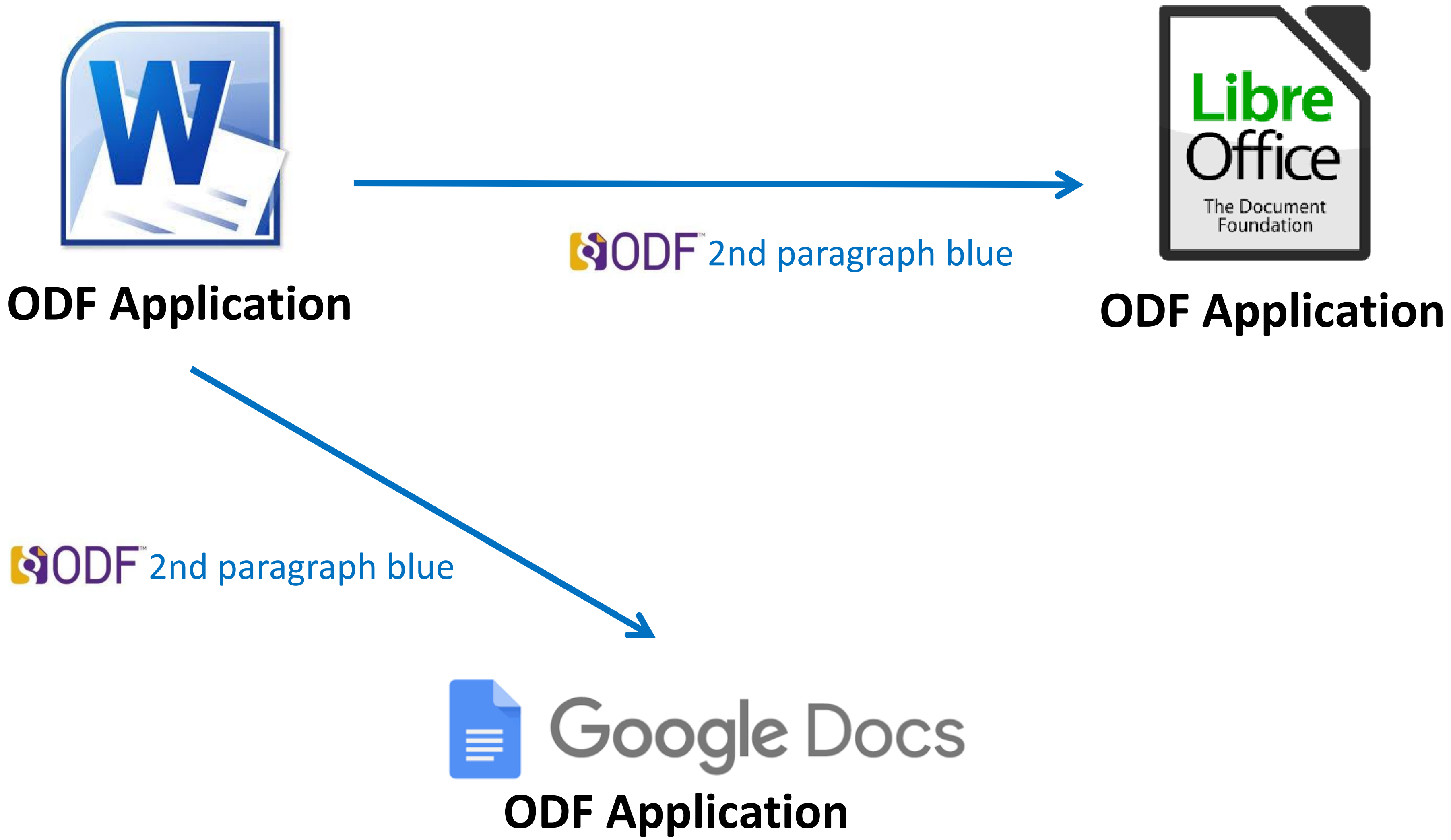
Exchanging ODF Changes



Multile users using different ODF applications exchanging no longer docs, but high level user changes! (standardized by OASIS being interoperable)

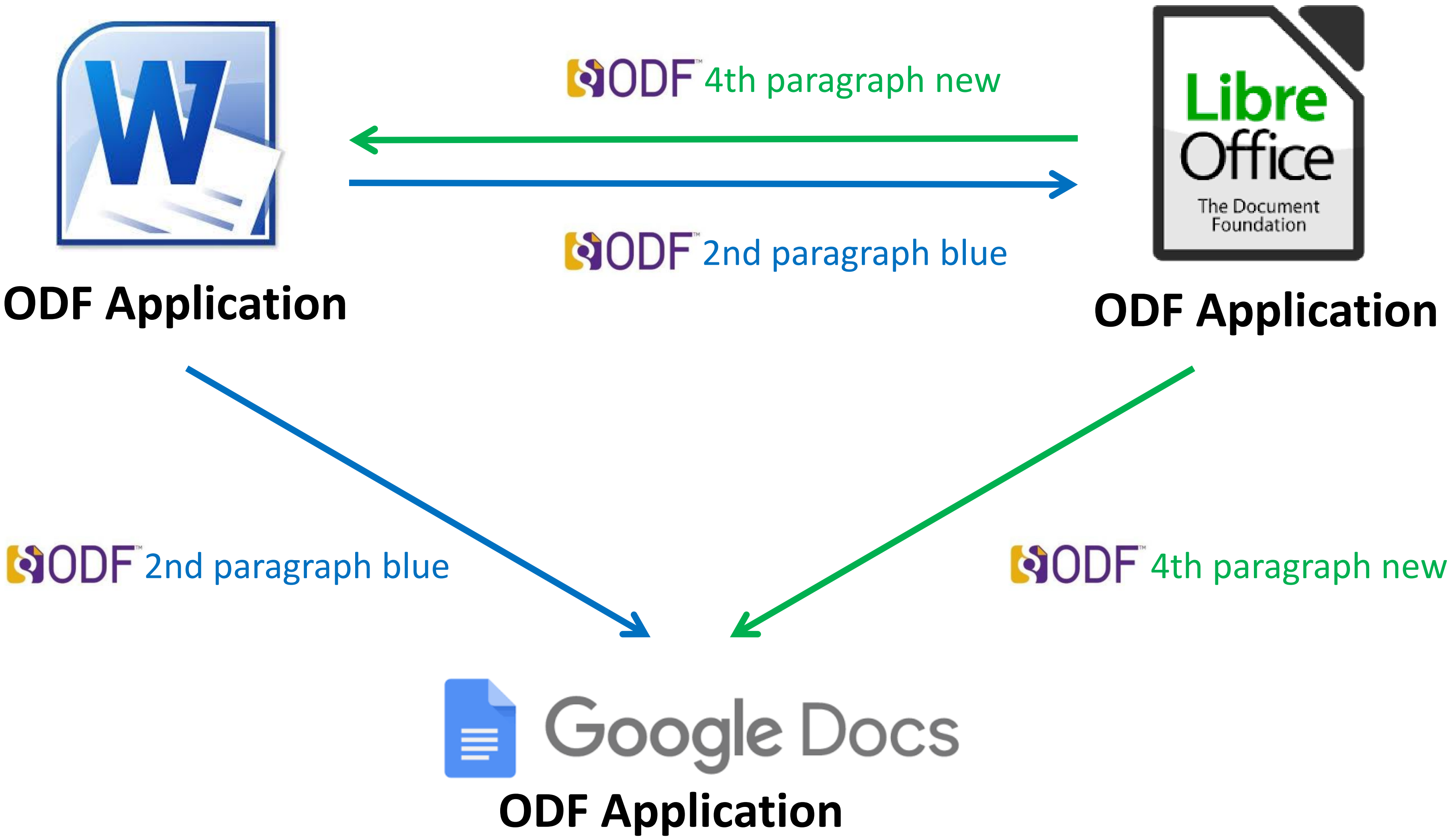
Interoperable Collaboration

Exchanging ODF Changes



Interoperable Collaboration

Exchanging ODF Changes



Are ODF Changes able to become a Standard?

LIBOCON TIRANA 2018

ODF Changes are de-facto Standard...

As all office application I am aware of...

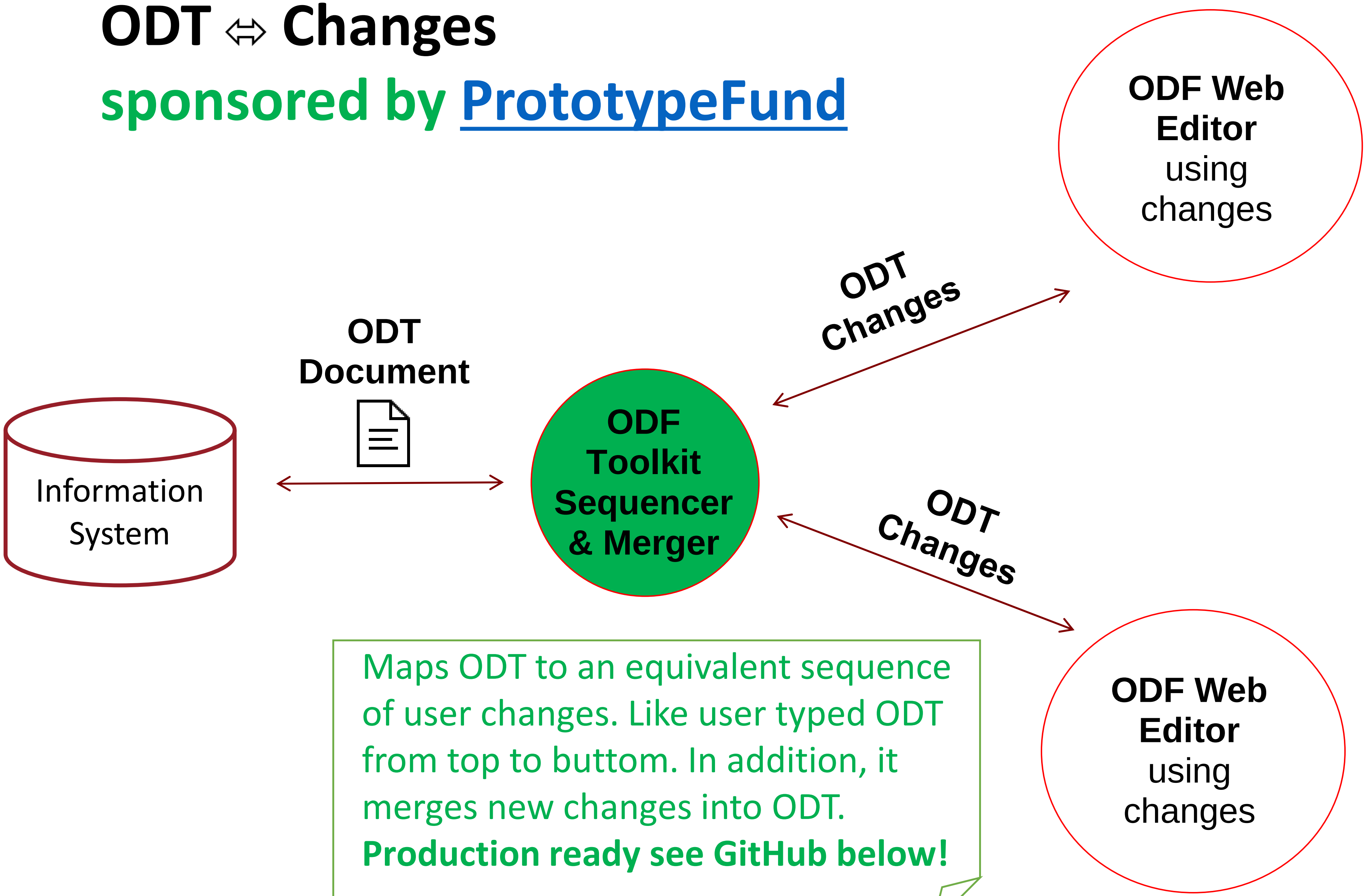
a) Know the same user objects (table, paragraph..)

b) Allow similar user changes (add, modify, delete …)

Just specifying what is already on our minds…

How about a Prototype on ODF Changes?

ODT ⇔ Changes
sponsored by PrototypeFund


ODF Web Editor using changes
ODT Changes
ODT Document
Information System
ODF Toolkit Sequencer & Merger
ODT Changes
ODF Web Editor using changes

Maps ODT to an equivalent sequence of user changes. Like user typed ODT from top to buttom. In addition, it merges new changes into ODT.
Production ready see GitHub below!

See https://github.com/svanteschubert/odftoolkit/tree/odf-changes/odfdom

LIBOCON TIRANA 2018

How about a Prototype an end user can use!?!

Most simple ODF application with only 2 feature is a text editor:

1) Every line is a paragraph

2) Text/characters without format

LIBOCON TIRANA 2018

Interoperable Collaboration

Exchanging ODF Changes

Vim still needs a way to record text position change & create operations!

ODF Application

ODF Application

ODF ADD „Hello " @1/1

ODF ADD „Hello " @3/1

ODF Feature Bridge

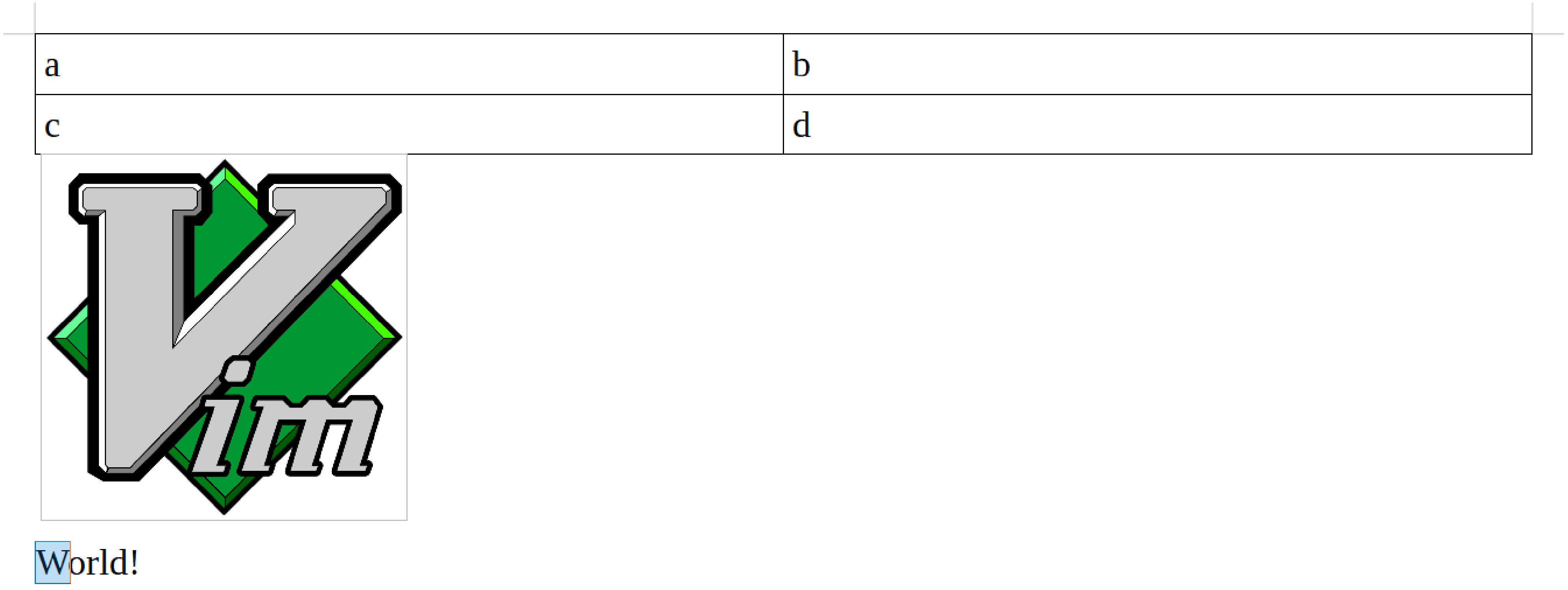
„Feature bridge" adds/removes changes of unsupported ODF features and adopts positions (OT), see above for LO has same change at 3rd, while VI at 1st position.

LIBOCON TIRANA 2018

Full Semantic Tree

Exchanging ODF Changes

Semantic tree:
The underlying XML tree is being mapped to larger logical pieces represented as Semantic Tree. Changes refer to those user objects.



a	b
c	d

World!

NOTE:

- VIM does not „see“ the table nor the image!
- The „**W**“ character is for **LibreOffice** at position „**3/1**“
- The „**W**“character is for **VIM** at position „**1/1**“

LIBOCON TIRANA 2018

Vim is <u>exciting</u>!

More thrills pls!!

Interoperable Collaboration

Exchanging ODF Changes

CKEditor 5
ODF Application

ODF Application

ODF ADD „Hello“ @3/1

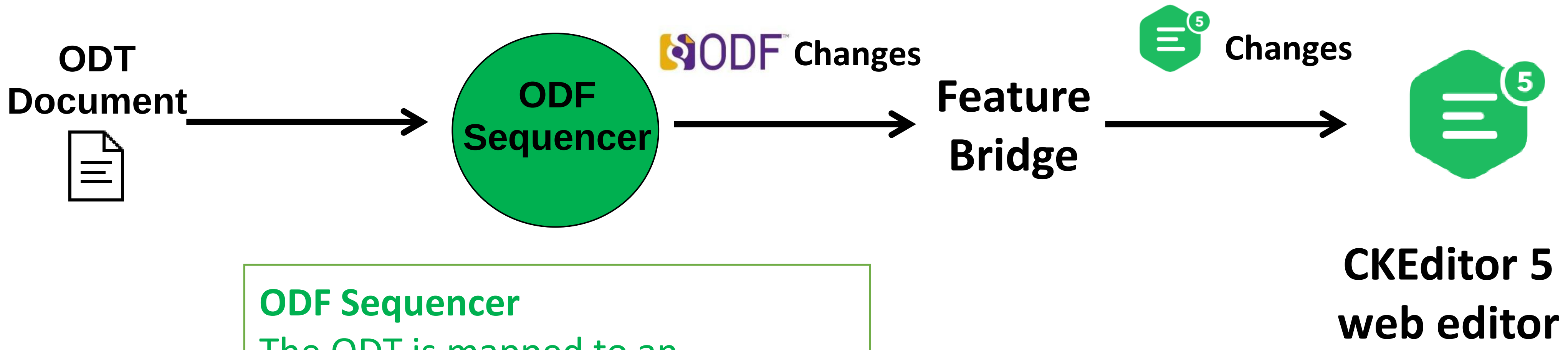
ODF ADD „Hello“ @3/1

ODF Feature Bridge

„Feature bridge“ not only adds/deletes changes, but maps them to other „change dialect“.
(more detailed view on next 2 slides)

Proof of Concept
Load ODF Text into CKEdit5

Exchanging ODF Changes

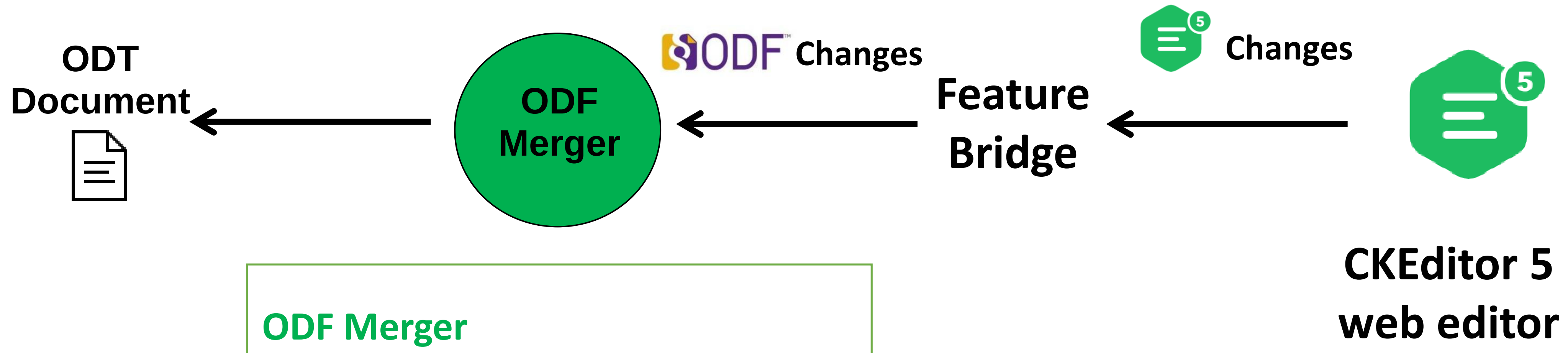


ODF Sequencer
The ODT is mapped to an equivalent list of user changes, as if a user had written the document from top to bottom.
(production ready)

LIBOCON TIRANA 2018

Proof of Concept
Save ODF Text by CKEdit5

Exchanging ODF Changes



ODF Merger
The new ODT user changes are merged into the document they are derived from.
(production ready)

Proof of Concept misses from CKEdit5:

a) Loading “CKEdit5 changes” from JSON
b) Saving new changes as “CKEdit5 changes” in JSON

Could someone help me with this, pls?!?
Thx in advance & cheers, Svante

Resources on CKEdit5 Changes

- here you will find all operations, with the inline documentation in the code

- here is the current version of the transformation (OT) code

- here you will find the engine debug plugin, which might be useful for debugging your code; all you need to do is to enable this plugin the same way you enable any other plugin and you should get some additional debug methods

- here you will find Operation Replayer; CKSource use it for debugging purposes to recreate the state of the model based on the recorded operation history (AFAIK not often recently used by CKSource)

- using "apply operation" event and method you should be able both record all operations applied to the document and apply your operations

What we learned so far:

1) Dispatching semantic changes is most efficient..
- **Changes are mandatory for merging**
- No longer heuristics required to find changes
- **Semantic provides best interoperability**

2) Changes perfect to **bridge different feature sets** of applications

New benefits (1/2)

- Save using Changes:
    - **No Data loss** by “Filters” overwriting unknown features
    - **Faster,** as only new changes are merged

- **Transparency** - **No fear of incidental overwriting data**
    - e.g. famous author receives change-request from reader

Key example: The read-only ODT of the famous author is accessible by changes counting positions. If instead we would use explicit IDs for position it would require to have an ID on all possible referenceable element blowing up the document size with IDs (boilerplate).
Convention over configuration!

New benefits (2/2)

- **Run Time API** across applications (based on **semantic tree**)
  - Browsers have Run Time API by W3C DOM
  - Semantic Tree is like a typed DOM ;-)

- Trustworthy **automated feature tests**

  Now: “Load doc” & “save doc”
  No proof, e.g. an array would support all ODF features

  Future: “Load doc”, “change some feature” & “save doc”

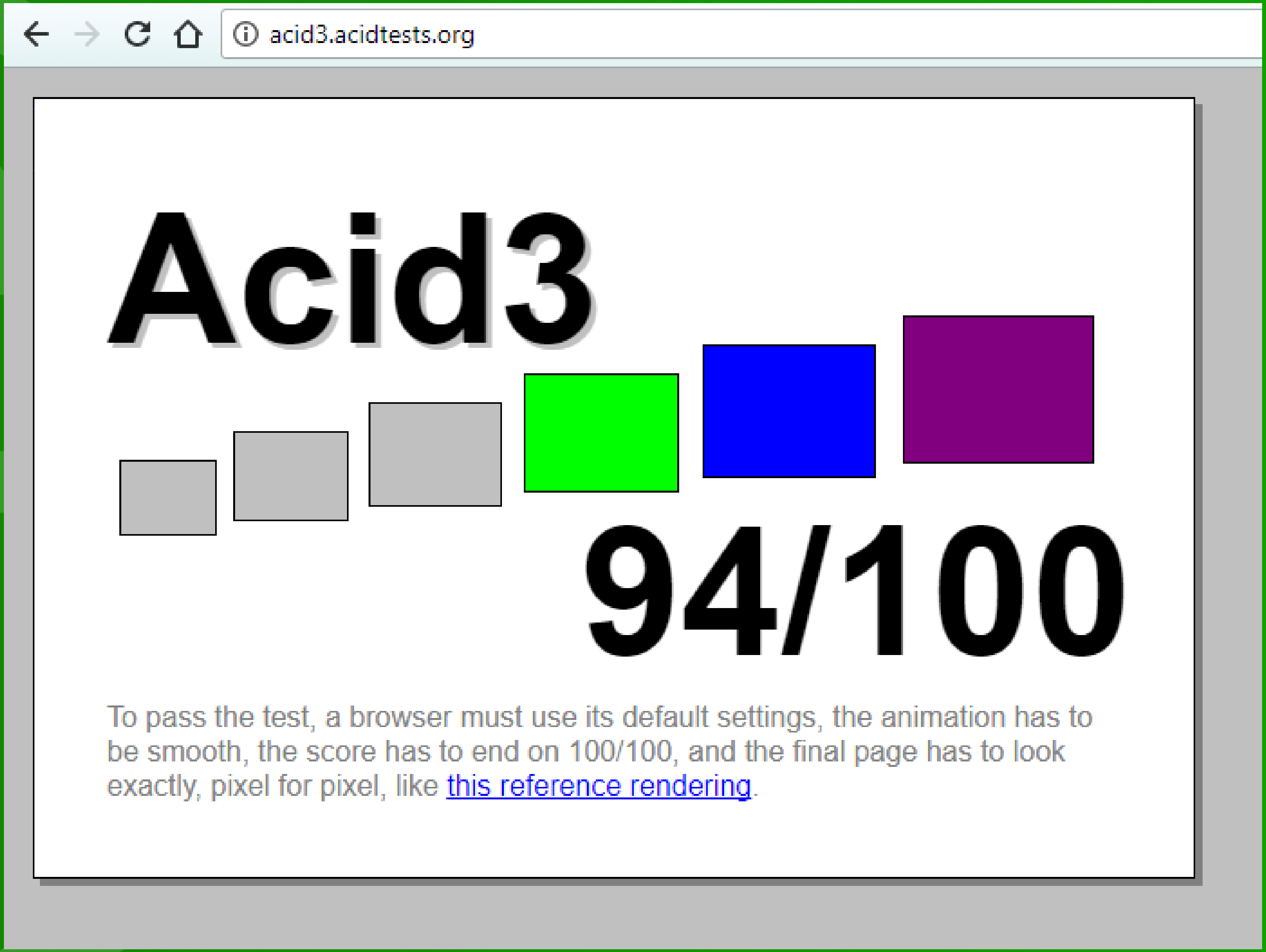
Q: ODF Run Time API?
What is it good for?

LIBOCON TIRANA 2018

A:Take a look at the browsers!

A: Interoperable Macros, similar JavaScript in Browsers!

Out of the box testing:

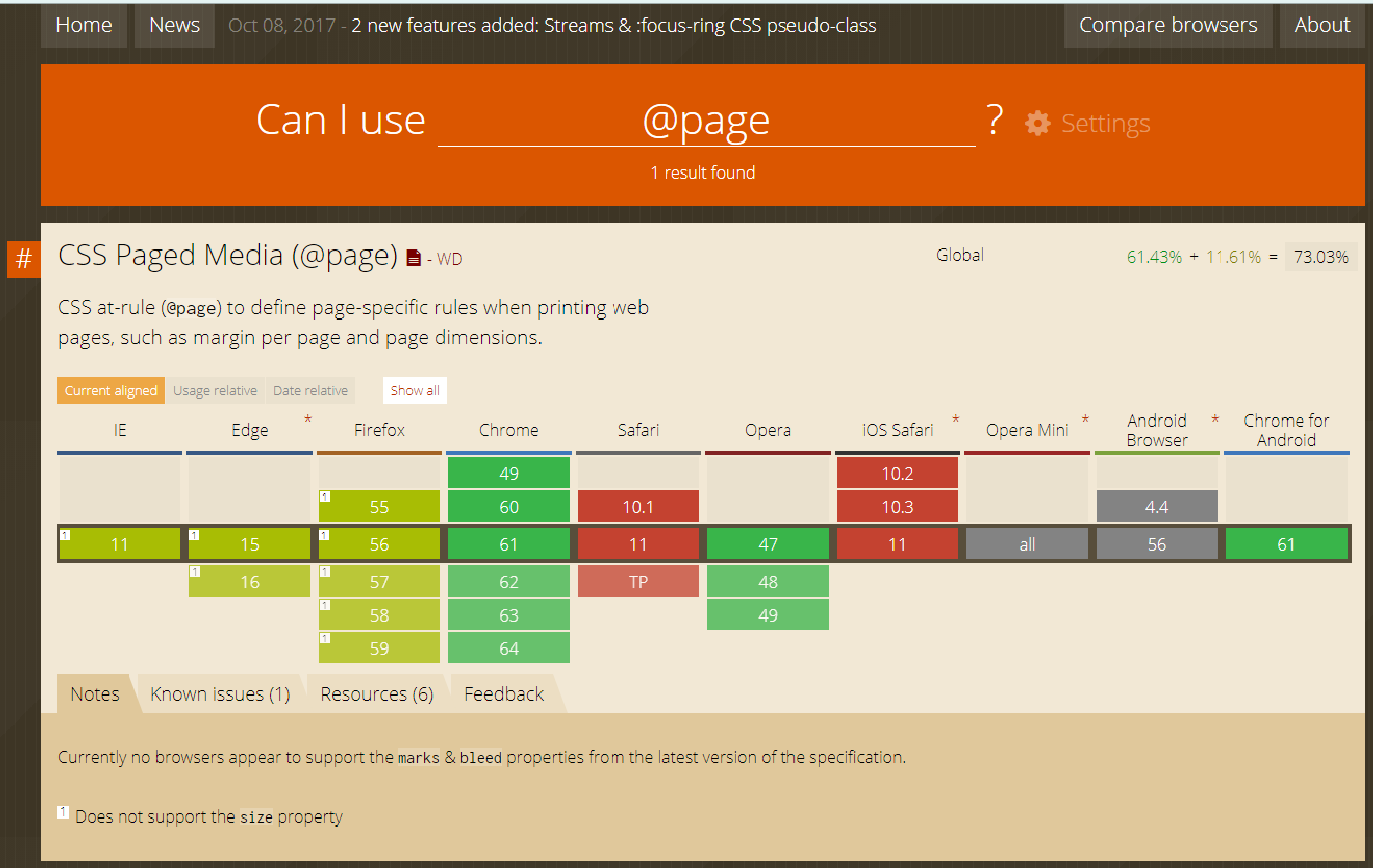

acid3.acidtests.org

Acid3

94/100

To pass the test, a browser must use its default settings, the animation has to be smooth, the score has to end on 100/100, and the final page has to look exactly, pixel for pixel, like this reference rendering.

Documentation on feature support:



Home | News | Oct 08, 2017 - 2 new features added: Streams & :focus-ring CSS pseudo-class | Compare browsers | About

Can I use @page ? Settings

1 result found

CSS Paged Media (@page) - WD

Global 61.43% + 11.61% = 73.03%

CSS at-rule (`@page`) to define page-specific rules when printing web pages, such as margin per page and page dimensions.

Current aligned | Usage relative | Date relative | Show all

IE	Edge *	Firefox	Chrome	Safari	Opera	iOS Safari *	Opera Mini *	Android Browser *	Chrome for Android
			49			10.2			
		55 [1]	60	10.1		10.3		4.4	
11 [1]	15 [1]	56 [1]	61	11	47	11	all	56	61
	16 [1]	57 [1]	62	TP	48				
		58 [1]	63		49				
		59 [1]	64						

Notes | Known issues (1) | Resources (6) | Feedback

Currently no browsers appear to support the `marks` & `bleed` properties from the latest version of the specification.

[1] Does not support the `size` property

https://caniuse.com/

LIBOCON TIRANA 2018

ODF app comparison based on features:

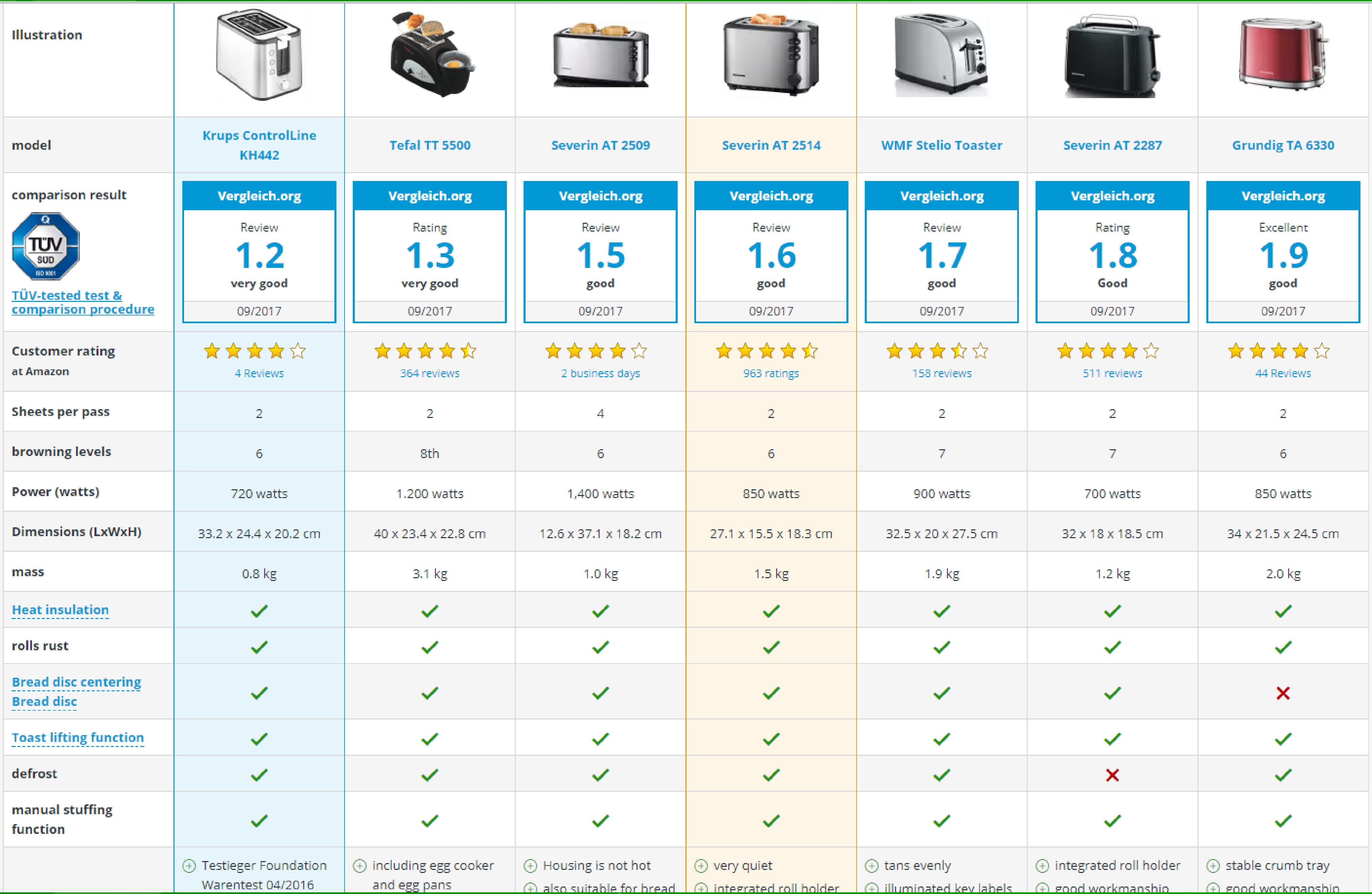


Illustration							
model	Krups ControlLine KH442	Tefal TT 5500	Severin AT 2509	Severin AT 2514	WMF Stelio Toaster	Severin AT 2287	Grundig TA 6330
comparison result TÜV-tested test & comparison procedure	Vergleich.org Review 1.2 very good 09/2017	Vergleich.org Rating 1.3 very good 09/2017	Vergleich.org Review 1.5 good 09/2017	Vergleich.org Review 1.6 good 09/2017	Vergleich.org Review 1.7 good 09/2017	Vergleich.org Rating 1.8 Good 09/2017	Vergleich.org Excellent 1.9 good 09/2017
Customer rating at Amazon	★★★★☆ 4 Reviews	★★★★⯪ 364 reviews	★★★★☆ 2 business days	★★★★⯪ 963 ratings	★★★⯪☆ 158 reviews	★★★★☆ 511 reviews	★★★★☆ 44 Reviews
Sheets per pass	2	2	4	2	2	2	2
browning levels	6	8th	6	6	7	7	6
Power (watts)	720 watts	1.200 watts	1,400 watts	850 watts	900 watts	700 watts	850 watts
Dimensions (LxWxH)	33.2 x 24.4 x 20.2 cm	40 x 23.4 x 22.8 cm	12.6 x 37.1 x 18.2 cm	27.1 x 15.5 x 18.3 cm	32.5 x 20 x 27.5 cm	32 x 18 x 18.5 cm	34 x 21.5 x 24.5 cm
mass	0.8 kg	3.1 kg	1.0 kg	1.5 kg	1.9 kg	1.2 kg	2.0 kg
Heat insulation	✓	✓	✓	✓	✓	✓	✓
rolls rust	✓	✓	✓	✓	✓	✓	✓
Bread disc centering Bread disc	✓	✓	✓	✓	✓	✓	✗
Toast lifting function	✓	✓	✓	✓	✓	✓	✓
defrost	✓	✓	✓	✓	✓	✗	✓
manual stuffing function	✓	✓	✓	✓	✓	✓	✓
	⊕ Testieger Foundation Warentest 04/2016	⊕ including egg cooker and egg pans	⊕ Housing is not hot ⊕ also suitable for bread	⊕ very quiet ⊕ integrated roll holder	⊕ tans evenly ⊕ illuminated key labels	⊕ integrated roll holder ⊕ good workmanship	⊕ stable crumb tray ⊕ good workmanship

Further benefits (1/2)

- Not only identifying the feature of applications:
  Also possible to **identify** the
  features of customer documents

Further benefits (2/2)

- **Git support** for ODF documents

    Overwrite GIT using semantic diffs instead of line based diffs.
    Standardized ODF changes the result of a
    comparison of two document!
    Merge will be so easy!!

Before you can understand: “Miracle of Merge”…

BASIC TECHNIQUES

The 1 x 1 of Changes / Operations

One Document –
Many ways to create it…

„ABC“

Final Document

One Document –
Many ways to create it...

User changes

ADD „A“ @1

„A“

Current Document

„ABC“

Final Document

One Document –
Many ways to create it...

User changes

Timeflow of changes

ADD „A“ @1
ADD „B“ @2

„AB“

Current document

„ABC“

Final document

LIBOCON TIRANA 2018

One Document –
Many ways to create it...

ADD „A“ @1
ADD „B“ @2
ADD „C“ @3

„ABC“

„ABC“

One Document –
Many ways to create it...

Different user changes

ADD „A“ @1
ADD „B“ @2
ADD „C“ @3

ADD „C“ @1

„C“

„ABC“

One Document –
Many ways to create it...

ADD „A“ @1
ADD „B“ @2
ADD „C“ @3

ADD „C“ @1
ADD „B“ @1

„BC“

„ABC“

One Document –
Many ways to create it...

ADD „A“ @1	ADD „C“ @1
ADD „B“ @2	ADD „B“ @1
ADD „C“ @3	ADD „A“ @1

„ABC“

„ABC“

One Document –
Many ways to create it...

ADD „A“ @1
ADD „B“ @2
ADD „C“ @3

⇔

ADD „C“ @1
ADD „B“ @1
ADD „A“ @1

QUESTION:

How transforming one into the other?🤔

„ABC“

One Document – Many ways to create it...

Move C change from top to bottom

ADD „A“ @1	**ADD „C“ @1**
ADD „B“ @2	ADD „B“ @1
ADD „C“ @3	ADD „A“ @1

„ABC“

LIBOCON TIRANA 2018

One Document – Many ways to create it...

Position of C changes as B was inserted now earlier, when the two changes are being switched!

ADD „A“ @1
ADD „B“ @2
ADD „C“ @3

ADD „B“ @1
ADD „C“ @2
ADD „A“ @1

„ABC“

One Document –
Many ways to create it...

ADD „A“ @1	ADD „B“ @1
ADD „B“ @2	ADD „A“ @1
ADD „C“ @3	**ADD „C“ @3**

„ABC“

LIBOCON TIRANA 2018

One Document – Many ways to create it...

Move B change from top to middle

ADD „A“ @1	**ADD „B“ @1**
ADD „B“ @2	ADD „A“ @1
ADD „C“ @3	ADD „C“ @3

„ABC“

One Document –
Many ways to create it…

Now both list of changes are identical!

ADD „A“ @1	ADD „A“ @1
ADD „B“ @2	**ADD „B“ @2**
ADD „C“ @3	ADD „C“ @3

We might call the blue list normalized!

„ABC“

Change Deletion

Can we delete B by just removing the change?

ADD „A“ @1
ADD „B“ @2
ADD „C“ @3

ADD „A“ @1
ADD „B“ @2
ADD „C“ @3

„A~~B~~C“

Change Deletion

NO deletion of a change from within the stack! Now a gap is in the positions, which is not allowed!

ADD „A“ @1
ADD „B“ @2
ADD „C“ @3

ADD „A“ @1
ADD „C“ @3

Error!!

„A~~B~~C“

LIBOCON TIRANA 2018

Change Deletion - Only remove top (last) change!

ADD „A“ @1
ADD „B“ @2
ADD „C“ @3

ADD „A“ @1
ADD „B“ @2
ADD „C“ @3

B not on top, not the last change being made, therefore B influences C

„A~~B~~C“

Change Deletion - Only remove top (last) change!

ADD „A“ @1
ADD „C“ @2
ADD „B“ @2

ADD „A“ @1
ADD „C“ @2
ADD „B“ @2

B last change, influences to C were removed by OT (see URL)

„A~~B~~C“

OT:
http://www.codecommit.com/blog/java/understanding-and-applying-operational-transformation

LIBOCON TIRANA 2018

Change Deletion - Only remove top (last) change!

ADD „A“ @1
ADD „C“ @2

⇔

ADD „A“ @1
ADD „C“ @2
ADD „B“ @2
DEL „B“ @2

Removes B and keep changes normalized.

Add inverse operation and keep all changes.

„AC“

LIBOCON TIRANA 2018

The Miracle of Merge

LIBOCON TIRANA 2018

Merging

USER A

ADD „Hello “ @1

„User A“ works on a branch (similar GIT branch concept) offline (e.g. sailing)

USER B

ADD „World “ @1

„User B“ works over week-end offline on own „git-like branch“. (e.g. woods)

Server state of ODT both user branched from

SERVER „“

LIBOCON TIRANA 2018

Merging

USER A

ADD „Hello “ @1

USER B

ADD „World “ @1

Push!

„User A“, first in office can PUSH to server

SERVER „“

Merging

USER A

ADD „Hello “ @1

USER B

ADD „World “ @1

Server state being adapted with branch of „User A“

ADD „Hello “ @1

„Hello “

SERVER

Merging

USER A

ADD „Hello “ @1

USER B

ADD „World “ @1

Push!

Error!!

„User B“ is not able to PUSH! Similar to software development with GIT „User B“ has to PULL first!

ADD „Hello “ @1

„Hello “

SERVER

Merging 1 / 5

USER A

ADD „Hello “ @1

USER B

ADD „World “ @1

User B pulls the change(s) being added earlier by „User A“ to its own branch.

Pull!

ADD „Hello “ @1

„Hello “

SERVER

Merging 2/5

USER A

ADD „Hello “ @1

USER B

ADD „World “ @1

ADD „Hello “ @1

Changes of „User A“ had happened BEFORE the changes of „User B“ and need to be moved to start of the change list, applying OT while moving.

ADD „Hello “ @1

„Hello “

SERVER

Merging 3/5

USER A

ADD „Hello “ @1

USER B

ADD „Hello “ @1

ADD „World “ @7

Position changes by OT from 1 to 7 due to the length of „Hello “

ADD „Hello “ @1

„Hello “

SERVER

Merging 4/5

USER A

ADD „Hello “ @1

USER B

ADD „Hello “ @1

ADD „World “ @7

Push!

ADD „World “ @7

ADD „Hello “ @1

„Hello “

SERVER

Now the new changes of „User B“ can be pushed as server is again on same base

Merging 5/5

USER A

ADD „Hello “ @1

USER B

ADD „Hello “ @1

ADD „World “ @7

NOTE:
If „User B“ would have pushed first ODT would be „World Hello „

ADD „Hello “ @1

ADD „World “ @7

„Hello World “

SERVER

Q: How do we identify documents (in GIT)?

A: Hash their Semantic Tree not the Syntax!

Get rid of „ODF XML syntax“ changing noise, breaking hashes identifying ODT documents in GIT!!

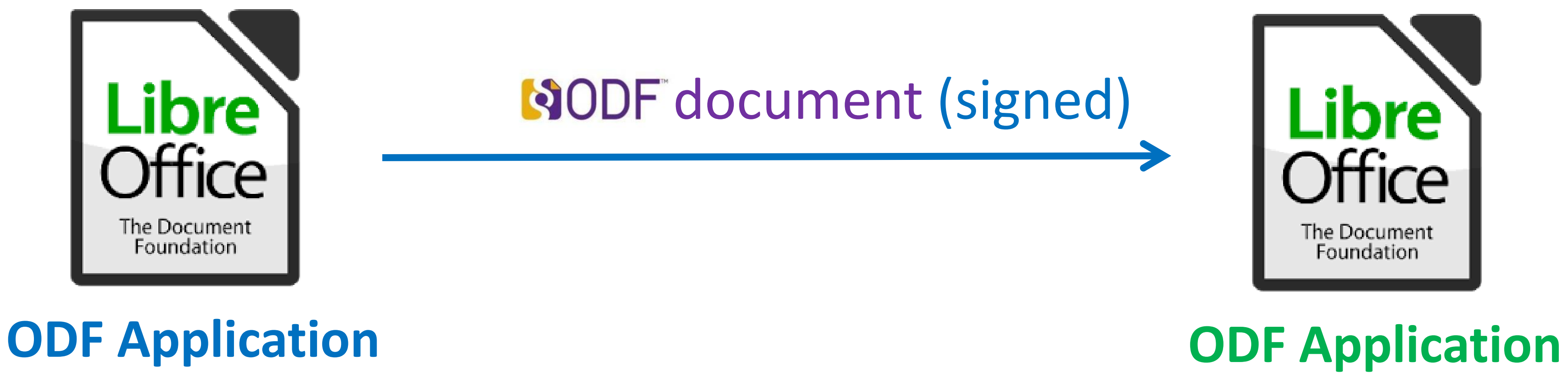
Semantic tree:
The underlying XML tree is being mapped to larger logical pieces represented as Semantic Tree. Changes refer to those user objects.

Q: How start the Collab feature in LibreOffice?

Q: What is the Minimum Viable Product (MVP)?

LO Collaboration (MVP)

Modern ping pong


Libre Office The Document Foundation
ODF document (signed)
Libre Office The Document Foundation
ODF Application
ODF Application

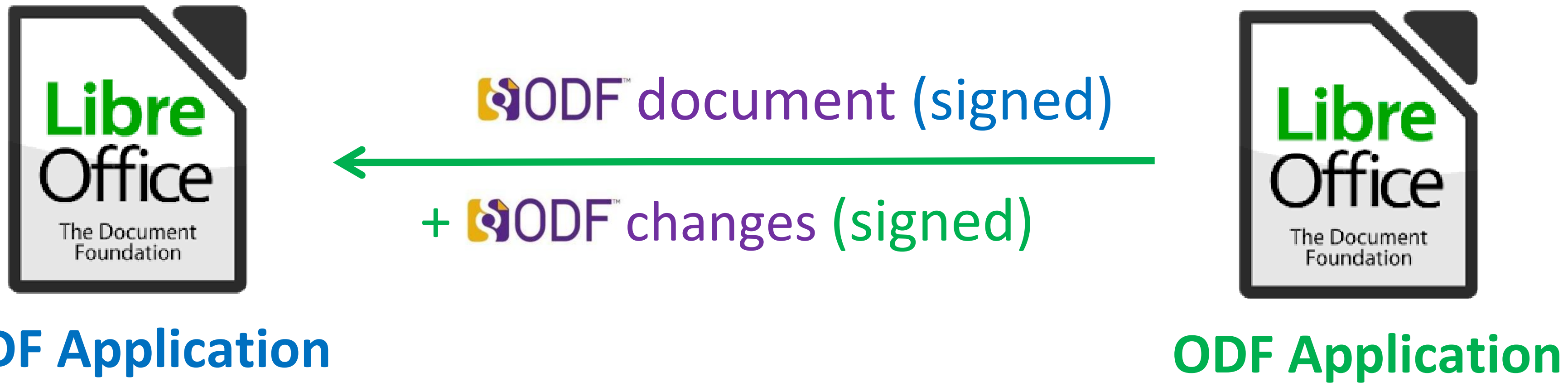
Like ancient wax seals, if you would change the content the seals break..

NOTE:
Not the ZIP of the ODT is being signed, but each XML file within the ZIP

But what if someone would like to suggest a change on a signed document?

LO Collaboration (MVP)

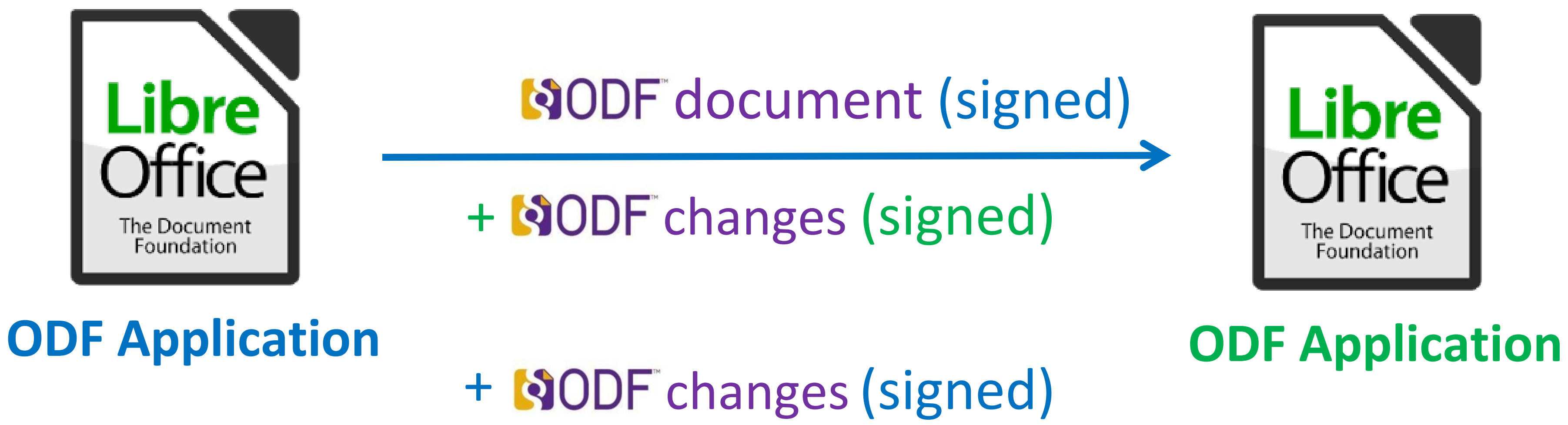
Modern ping pong



- Suggested changes could be saved within the ODT ZIP as a new file, pointing to the content!
- By this the XML sign of the content.xml file would not be broken!
- New file could be signed as well with the signature of the responding user!

LO Collaboration (MVP)

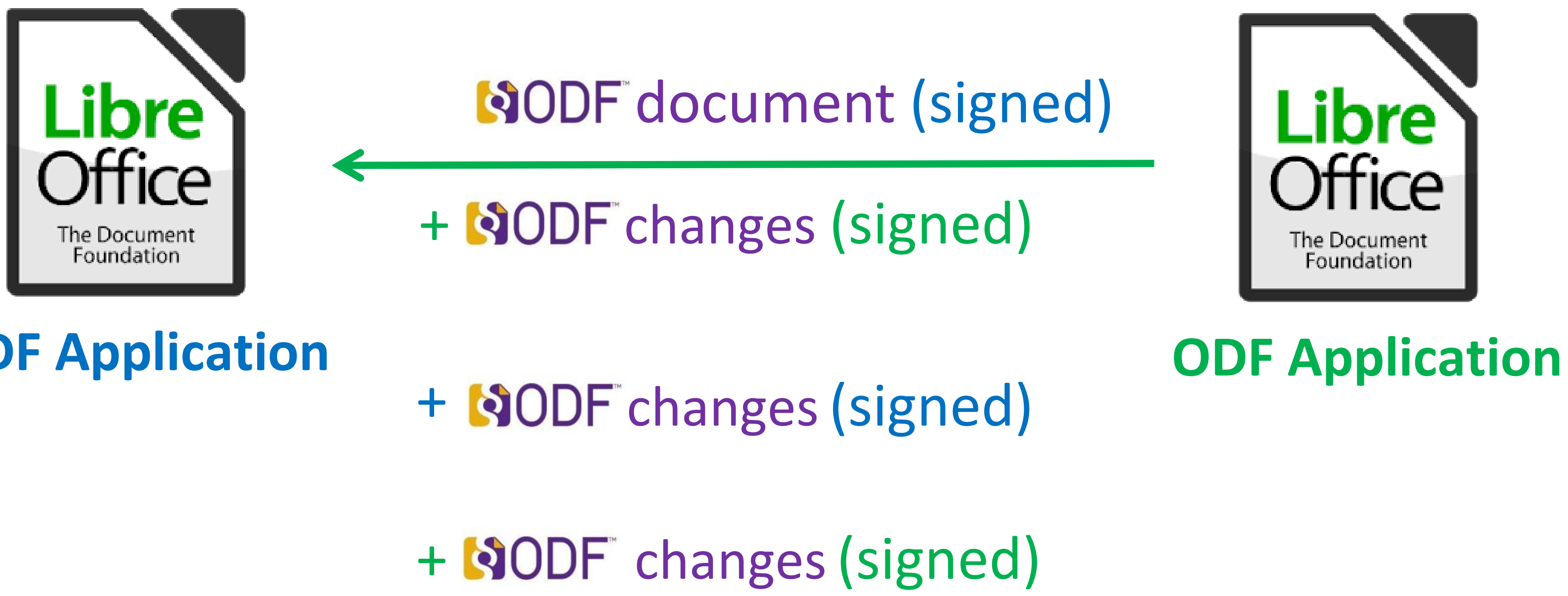
Modern ping pong



- Initial author can still answer by also saving new changes!
- Although all changes will be kept, earlier suggested changes can be removed by adding their inverse change!

LO Collaboration (MVP)

Modern ping pong



- Authentication of every change of any editor in document history is being guaranteed!

When it is so cool,

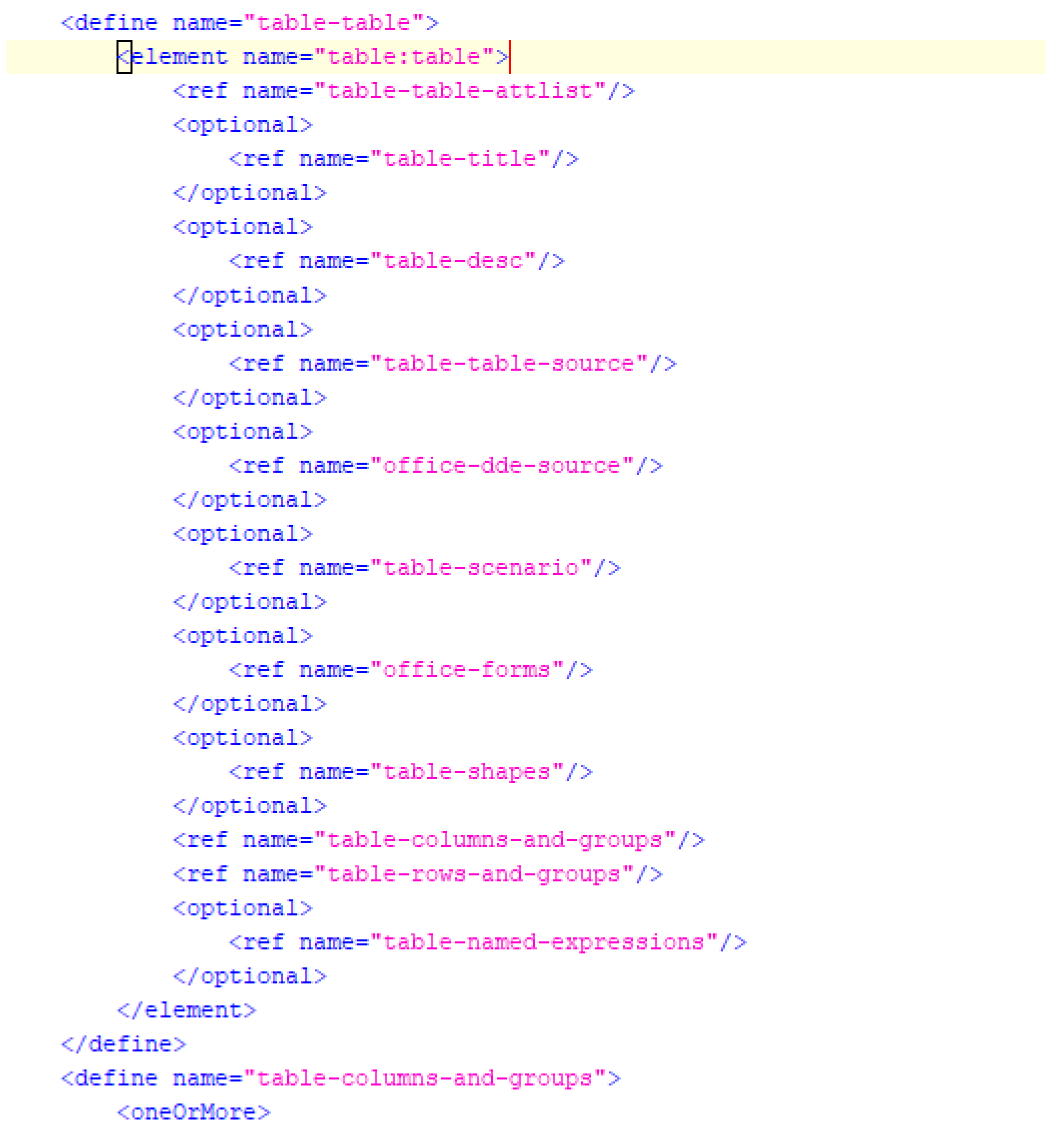
why don’t we have it already?

ODF Changes based on

ODF XML, which is complex..

ODF XML Grammar

Hard to oversee



```
<define name="table-table">
    <element name="table:table">
        <ref name="table-table-attlist"/>
        <optional>
            <ref name="table-title"/>
        </optional>
        <optional>
            <ref name="table-desc"/>
        </optional>
        <optional>
            <ref name="table-table-source"/>
        </optional>
        <optional>
            <ref name="office-dde-source"/>
        </optional>
        <optional>
            <ref name="table-scenario"/>
        </optional>
        <optional>
            <ref name="office-forms"/>
        </optional>
        <optional>
            <ref name="table-shapes"/>
        </optional>
        <ref name="table-columns-and-groups"/>
        <ref name="table-rows-and-groups"/>
        <optional>
            <ref name="table-named-expressions"/>
        </optional>
    </element>
</define>
<define name="table-columns-and-groups">
    <oneOrMore>
```

<u>ODF 1.2 XML:</u>

. 598 XML elements

. 1300 XML attributes

>18 tsd. lines

Very hard to read by humans and to search within!

ODF XML Grammatik

Hard to oversee

```
<define name="table-table">
  <element name="table:table">
        <ref name="table-table-attlist"/>
        …

        <optional>
            <ref name="text-soft-page-break"/>
        </optional>
        <ref name="table-table-row"/>
```

ODF 1.2 XML:

. 598 XML elements

. 1300 XML attributes

>18 tsd. lines

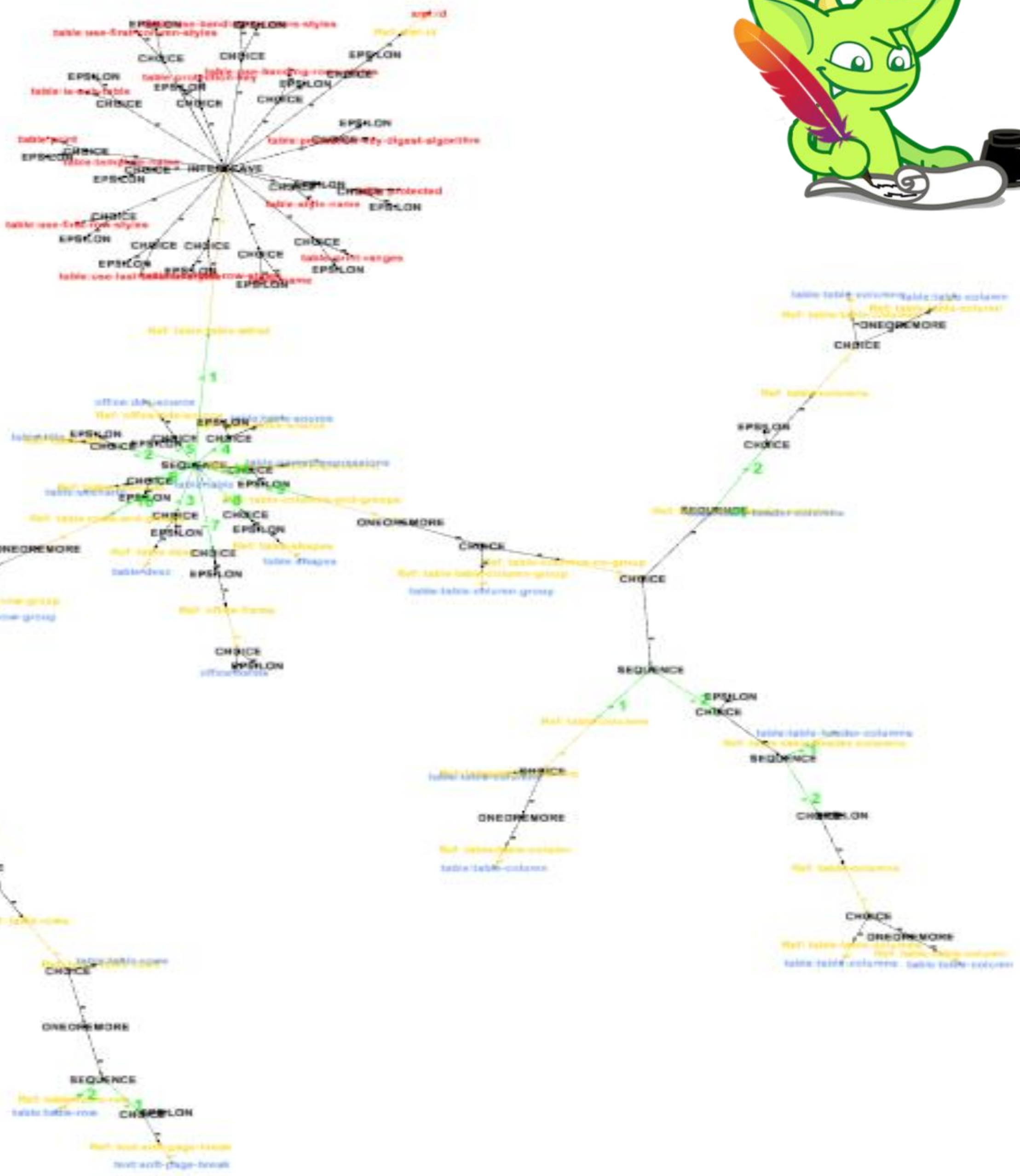
Let‘s look only at the <table:table> root element and its children in the ODF XML grammar.

LIBOCON TIRANA 2018

ODF Grammar - Graph

Table root with children

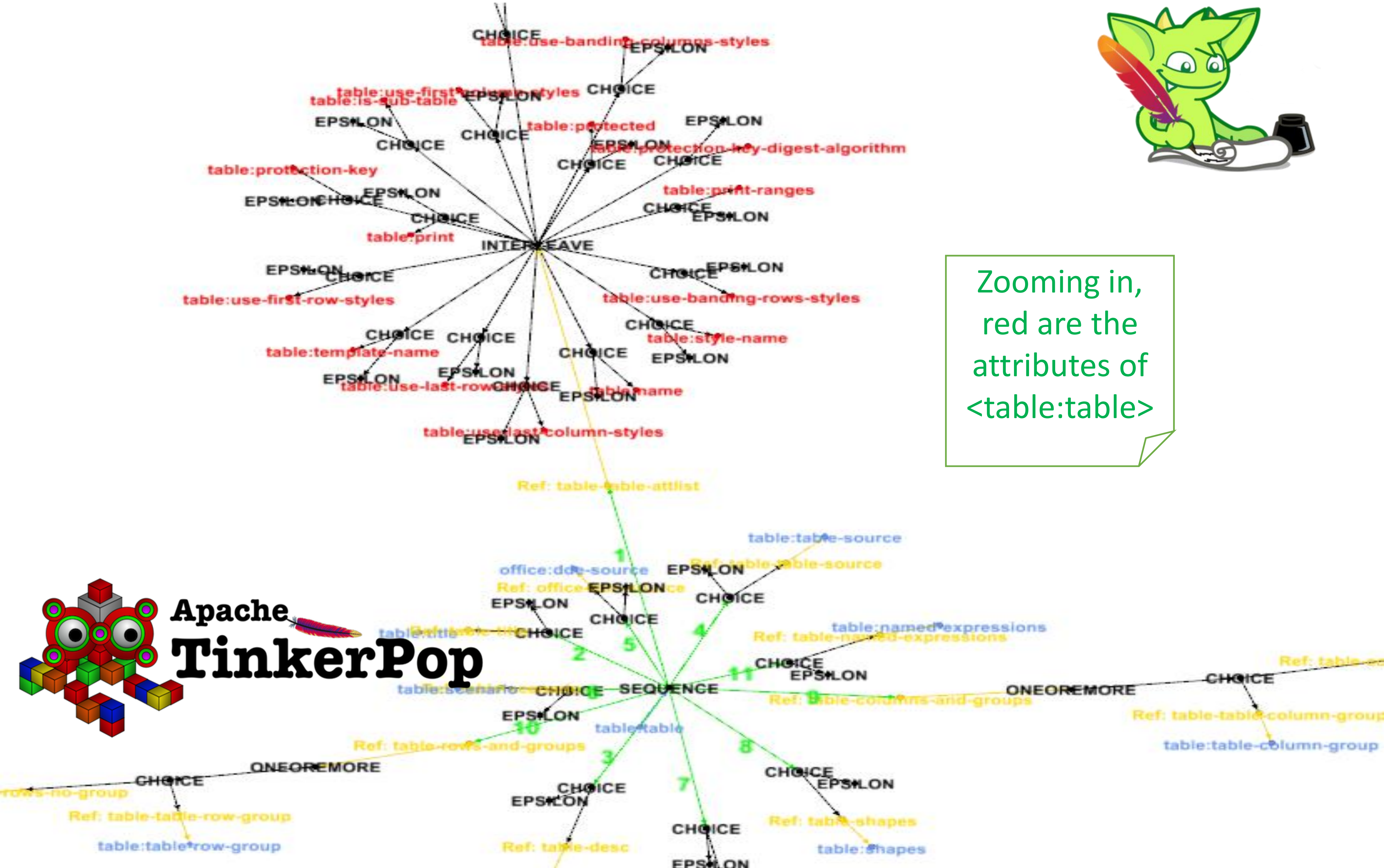
Table root with children part of ODF grammar loaded into Graph DB and visualized with Gelphi



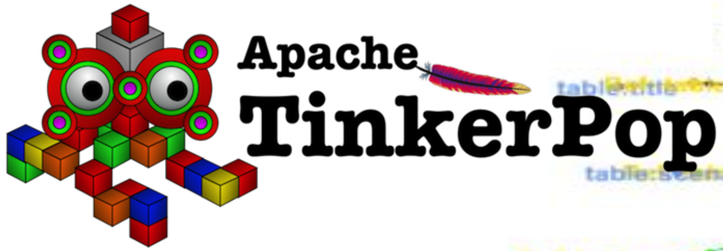
Apache TinkerPop

ODF Grammar - Graph

Table root with children

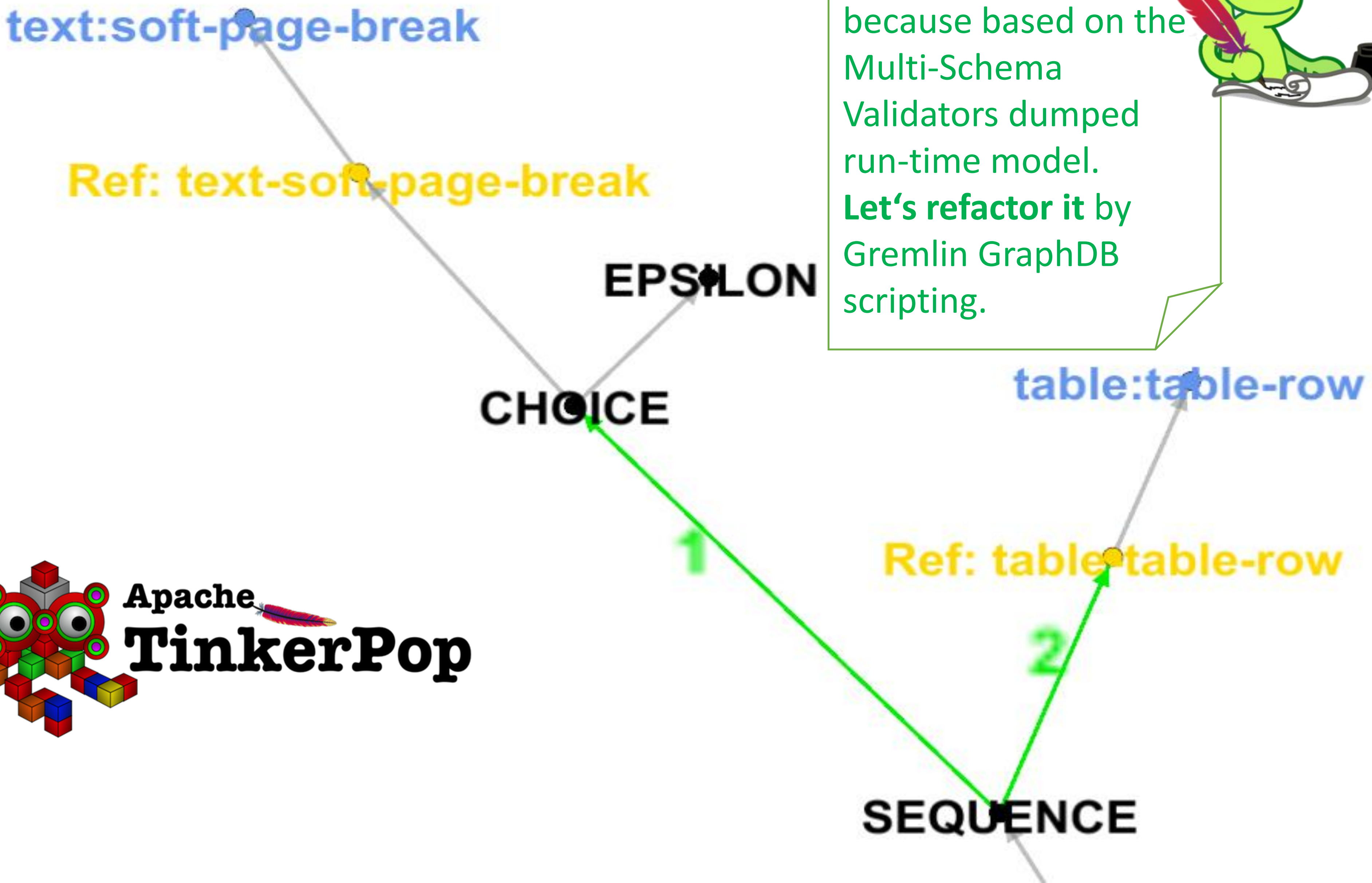


Zooming in, red are the attributes of <table:table>



ODF Grammar - Graph

Cumbersome

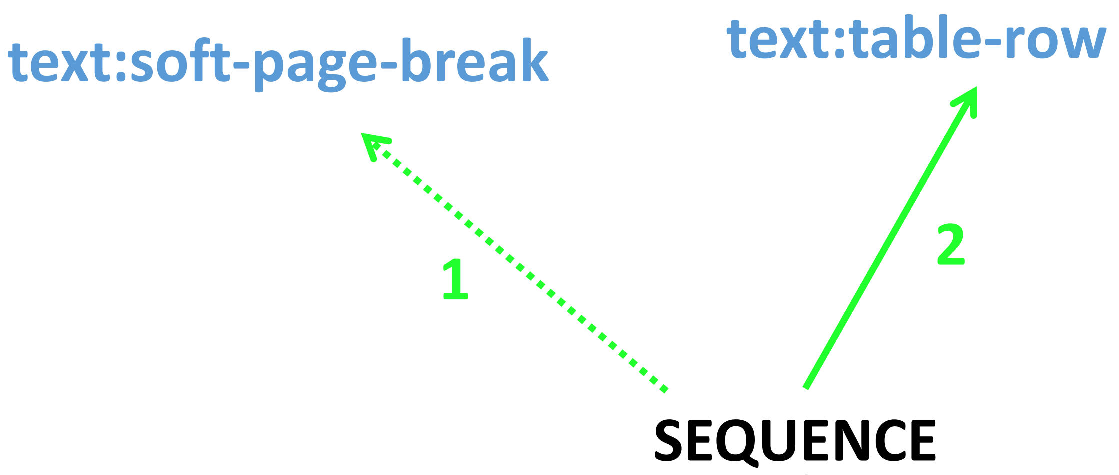


Graph still complex because based on the Multi-Schema Validators dumped run-time model. **Let's refactor it** by Gremlin GraphDB scripting.

ODF Grammar - Graph

SIMPLIFIED

Same semantic as slide before but refactored for better human understanding



NOTE:
Graph DB allows queries as „can a <text:p> paragraph element be nested, find out far easier and reproduciable instead of looking up 18k of lines of grammar.

GOAL & VISION (1/2)

a) Define additional ODF Change info!
b) Ease access to ODF XML grammar via
Graph DB

GOAL & VISION (2/2)

c) From above: Generate source code ->
ODF RunTimeModel

d) Let become <u>collab editors</u>
as frequent as <u>text editors</u>!

Huge number of ODF XML should be taggled by source code generation.
More flexible to create RunTimeModl with for different languages! Or optimization such bitarrays for spreadsheet cells properties.

CURRENTLY I AM:

a) Generating Source Code for eInvoice EU standard
b) Love to elaborate the collab idea with <u>YOU</u>!

No one can tell if LibreOffice is still en vogue in 50 years!

But the **collaboration feature** is **critical**!

Q&A anytime!

BY SA

All text and image content in this document is licensed under the Creative Commons Attribution-Share Alike 4.0 License (unless otherwise specified)

LIBOCON TIRANA 2018