Office in Java

2. Info-Point

Urs Frei

Problemstellung:

?

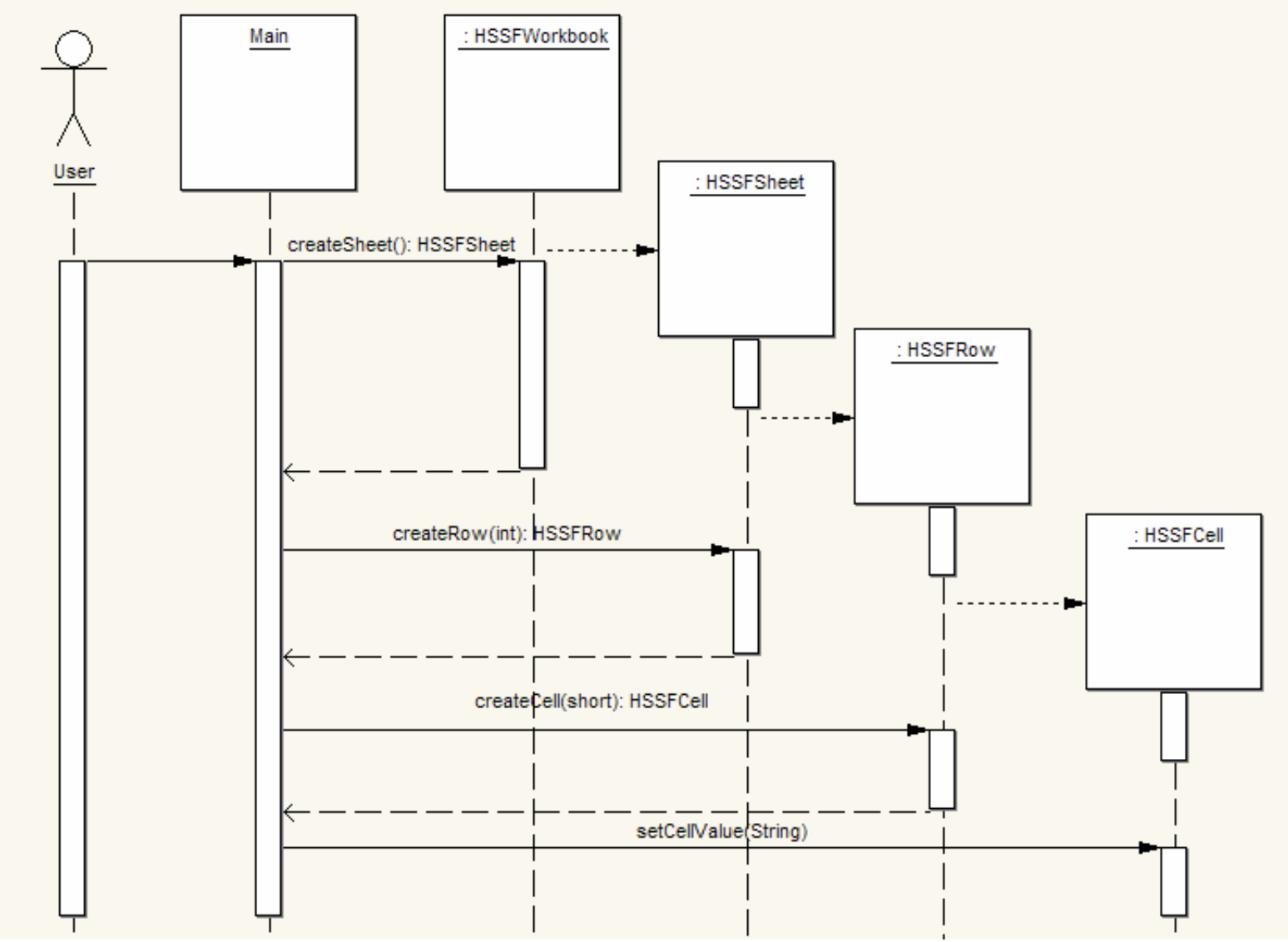
Zwei Motivationen

- Daten direkt im richtigen Format
  - Excel nicht über Zwischenformat csv
  - In Word Bereichte erstellen
- Office fernsteuern
  - Rechtschreibeprüfung Word
  - Rechnungsfunktionen Excel
  - Adressen von Outlook

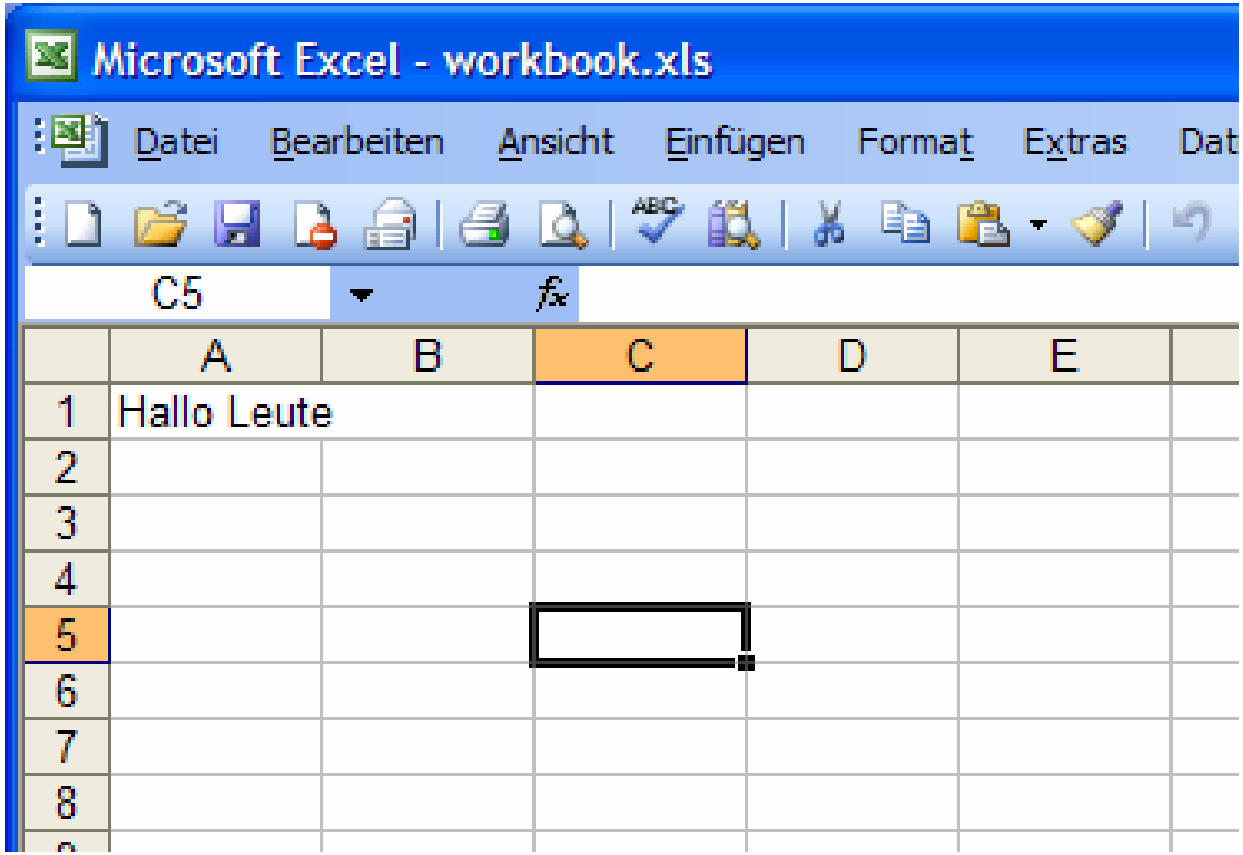
Jakarta POI

- http://jakarta.apache.org/poi/
- Framework zum erstellen von Office Files. Besteht aus:
  - HSSF → Excel (97 bis 2002)
  - HWPF → Word (97 bis 2002) frühes Entwicklungsstadium noch nicht verfügbar (nur cvs)

Bsp. mit HSSF


User
Main
: HSSFWorkbook
: HSSFSheet
: HSSFRow
: HSSFCell
createSheet(): HSSFSheet
createRow(int): HSSFRow
createCell(short): HSSFCell
setCellValue(String)

Bsp. Resultat



Was HSSF nicht kann (Excel)

- Unterschiedliche Formatierungen in der gleichen Zelle
- Keine Diagramme erzeugen
- Keine Makro erzeugen
- Pivot Tabelle können nicht erzeugt werden
- http://jakarta.apache.org/poi/hssf/quick-guide.html

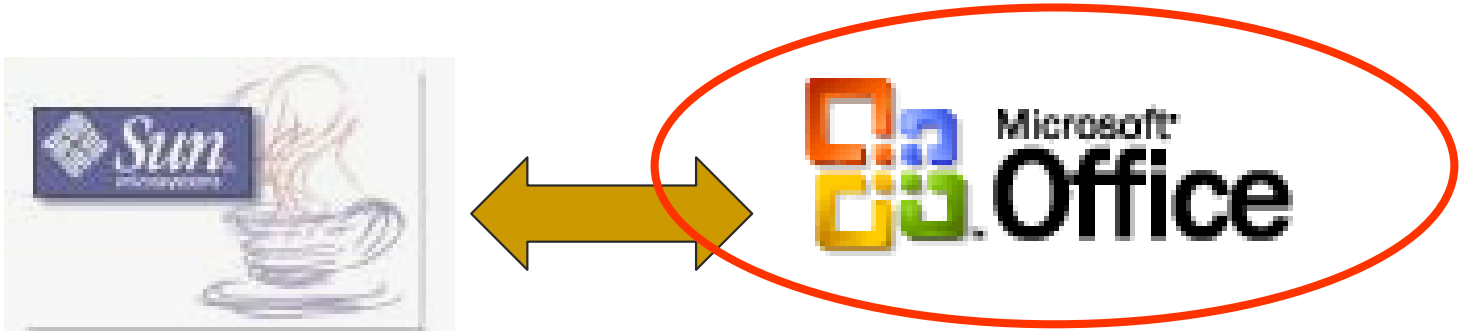
Was HWPF nicht kann (Word)

- Einfacher was es kann
  - Offiziell noch nichts ☹
- Erst CVS Version erhältlich
- Es werden Entwickler gesucht.

Zwei Motivationen

- Daten direkt im richtigen Format
  - Excel nicht über Zwischenformat csv
  - In Word Bereichte erstellen
- Office fernsteuern
  - Windowsprogramme nutzbar machen
    - Rechtschreibeprüfung Word
    - Rechnungsfunktionen Excel
    - Adressen von Outlook

Windowsarchitektur



- Component Object Model (COM)
  - Automatisation
  - OLE
  - ActiveX
  - DCOM
- Office ist als COM Implementiert

Was ist COM (1)

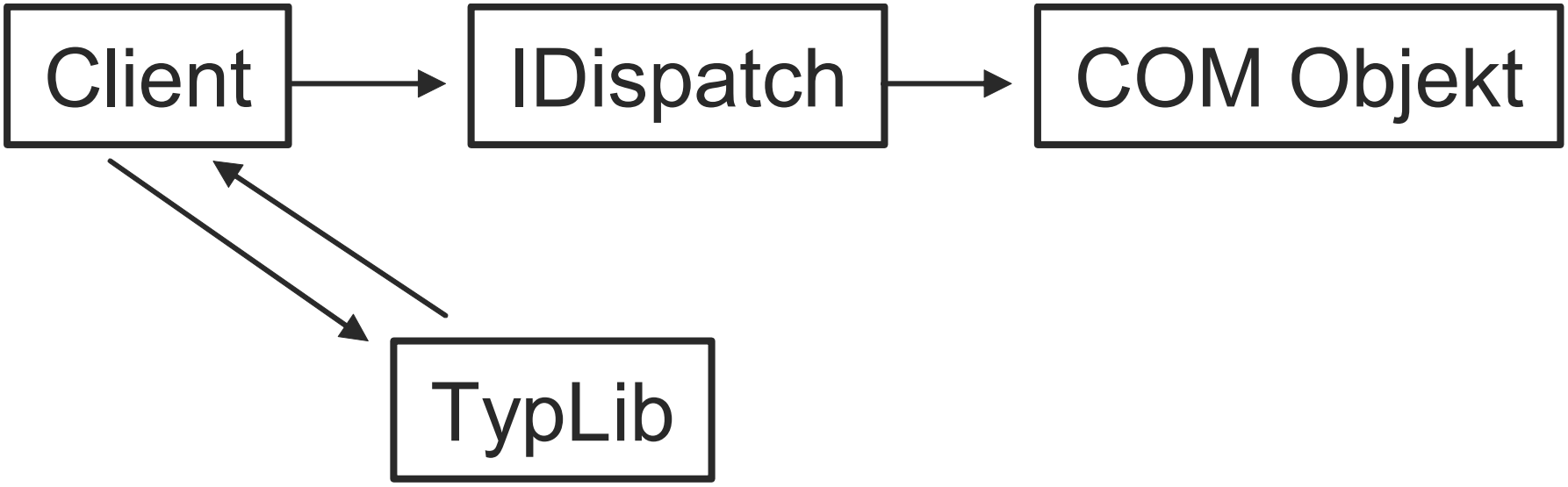
- Komponenten-Architektur
- Bsp. Word ist aus mehrer COM Komponenten aufgebaut
  - Dokument
  - Textmarken
  - Tabellen
- Fernsteuern von Anwendungen

Was ist COM (2)

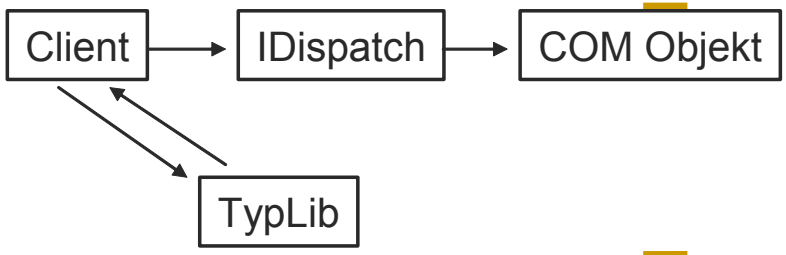
- Ähnlich Reflection von Java
- Typprüfung erfolgt zur Laufzeit
- Komponente kann als OLE Server oder als OLE Client agieren
- Ermöglicht einbetten von Elemente
  - Bsp. Excel Tabelle in Word
- Üblicherweise bei Registry registriert

Funktionsweise

- Zusammenspiel der COM Bestandteile

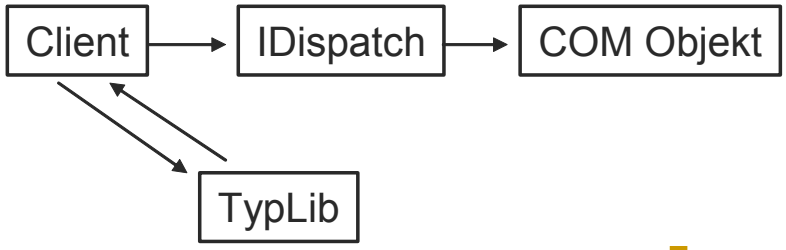


IDispatch



- Wird von allen COM Objekten implementiert
- Stellt „primitive“ Zugriffsmöglichkeiten auf Objekt zur Verfügung
  - Invoke (wie Reflection) Parameter definiert welche Methode auf COM aufgerufen wird

TypeLib



- Darin ist definiert welche Funktionalität das COM Objekt unterstützt
- Diese Funktionalität wird über die Methode Invoke des IDispach Intefaces verwendet
- Div. Erweiterungen: tlb, olb, dll, ocx, exe, usw.

Einfaches Beispiel in VB

- Word öffnen
- Word anzeigen
- Neues Dokument erstellen
- Text in das Dokument schreiben

Umsetzung mit Visual Basic (späte Bindung)

```
Dim document As Object
Dim word As Object
Dim documentCont As Object
Dim newDocument As Object
Dim newRange As Object

Set document = CreateObject("Word.Document")
Set word = document.application
word.Visible = True
Set documentCont = word.Documents
Set newDocument = documentCont.Add
Set newRange = newDocument.Range
newRange.Text = "Hallo Leute"
```

Erklärung Bsp. (1)

- CreateObject("Word.Document")
  - Parameter in Registry nachschlagen
    - Über TypeLib MSWord.olb die uuid in der Registry suchen um ProgID zu finden
    - Oder über Dateierweiterung, die das Programm verwendet (.doc)

Erklärung Bsp. (2)

- Wie heissen Methoden?
  - Informationen aus msword.olb (ist TypLib File)

```
interface _Document : IDispatch {
    [id(00000000), propget, helpcontext(0x096b0000)]
    HRESULT Name([out, retval] BSTR* prop);
    [id(0x00000001), propget, helpcontext(0x096b0001)]
    HRESULT Application([out, retval] Application** prop);
    [id(0x000003e9), propget, helpcontext(0x096b03e9)]
    HRESULT Creator([out, retval] long* prop);
    [id(0x000003ea), propget, helpcontext(0x096b03ea)]
    HRESULT Parent([out, retval] IDispatch** prop);
    [id(0x000003e8), propget, helpcontext(0x096b03e8)]
    HRESULT BuiltInDocumentProperties([out, retval] IDispatch** prop);
    [id(0x00000002), propget, helpcontext(0x096b0002)]
    HRESULT CustomDocumentProperties([out, retval] IDispatch** prop);
    [id(0x00000003), propget, helpcontext(0x096b0003)]
    HRESULT Path([out, retval] BSTR* prop);
```

Wie von Java auf COM?

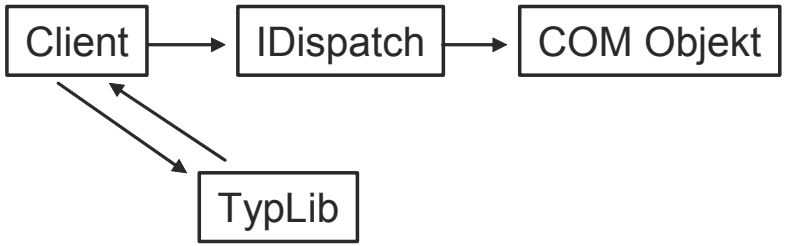
- JNI selber implementieren
  - Sehr aufwendig
- SWT
  - Abstrahiert JNI (immer noch aufwendig)

Was ist SWT

- Framework zur Darstellung von Java Applikationen
- Vergleichbar mit AWT
- Setzt auf Betriebsystemkomponenten auf
- Bestandteil von Eclipse

Und dies alles nur für etwas Text in Word….

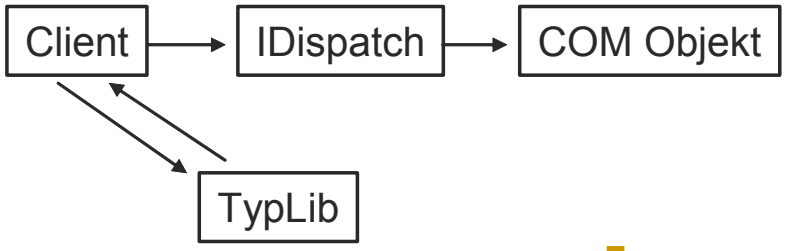
```
OleFrame oleFrame = new OleFrame(new Shell(), SWT.NONE);
OleClientSite controlSite = new OleClientSite(oleFrame, SWT.NONE, "Word.Document");
OleAutomation document = new OleAutomation(controlSite);

int[] methodID = document.getIDsOfNames(new String[]{"Application"});
Variant vApplication = document.getProperty(methodID[0]);
methodID = vApplication.getAutomation().getIDsOfNames(new String[]{"Visible"});
vApplication.getAutomation().setProperty(methodID[0], new Variant(true));
methodID = vApplication.getAutomation().getIDsOfNames(new String[]{"Documents"});

Variant vDocumentCont = vApplication.getAutomation().getProperty(methodID[0]);
methodID = vDocumentCont.getAutomation().getIDsOfNames(new String[]{"add"});

Variant vNewDocument = vDocumentCont.getAutomation().invoke(methodID[0]);
methodID = vNewDocument.getAutomation().getIDsOfNames(new String[]{"Range"});

Variant vNewRange = vNewDocument.getAutomation().invoke(methodID[0]);
methodID = vNewRange.getAutomation().getIDsOfNames(new String[]{"Text"});
vNewRange.getAutomation().setProperty(methodID[0], new Variant("Hallo Leute"));
```



SWT Verwendung(1)

```
OleFrame oleFrame = new OleFrame(new Shell(), SWT.NONE);
OleClientSite controlSite = new OleClientSite(oleFrame, SWT.NONE, "Word.Document");
OleAutomation document = new OleAutomation(controlSite);
```

- Erzeugt das COM Object
- Alle COM Objecte sind vom Type OleAutomation



SWT Verwendung(2)

```
int[] methodID = document.getIDsOfNames(new String[]{"Application"});
Variant vApplication = document.getProperty(methodID[0]);
```

- Methoden usw. können nicht über Namen aufgerufen werden
- Methodenaufruf erfolgt über ID der Methode
  - ID kann dem TypLib File entnommen werden
  - Oder über getIDsOfNames des OleAutomation Objekts angefragt werden

SWT Verwendung(3)

- Entscheidung:
  - Methodenaufruf?
    - invoke
  - Property setzen?
    - setProperty
  - Property abrufen?
    - getProperty

SWT Verwendung (4)

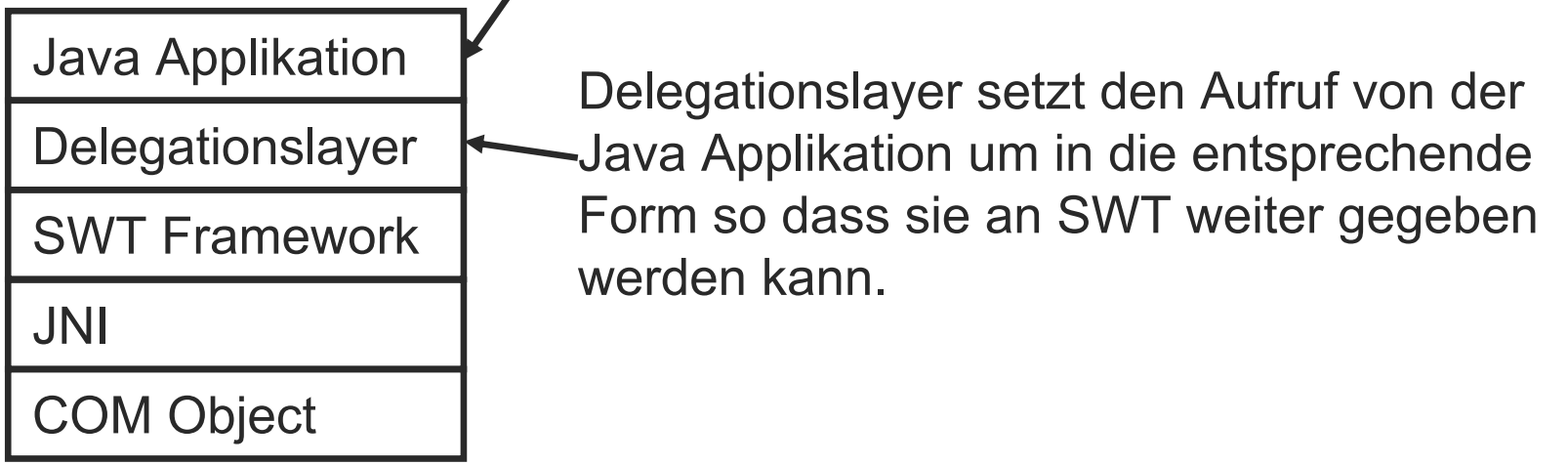
```
Variant vNewRange = vNewDocument.getAutomation().invoke(methodID[0]);
methodID = vNewRange.getAutomation().getIDsOfNames(new String[]{"Text"});
vNewRange.getAutomation().setProperty(methodID[0], new Variant("Hallo Leute"));
```

- Universaldatentyp „Variant“
  - Return-Wert Übergabeparameter
  - Vergleichbar mit Object in Java

Zwischenschicht für akzeptable Verwendung

- SWT im Vergleich zu VB viel aufwendiger
  → Architektur soll Aufwand reduzieren
- Mittels Delegation Aufruf vereinfachen
  → COM Document Objekt hat entsprechendes Document Objekt in Java

Architektur Zwischenschicht



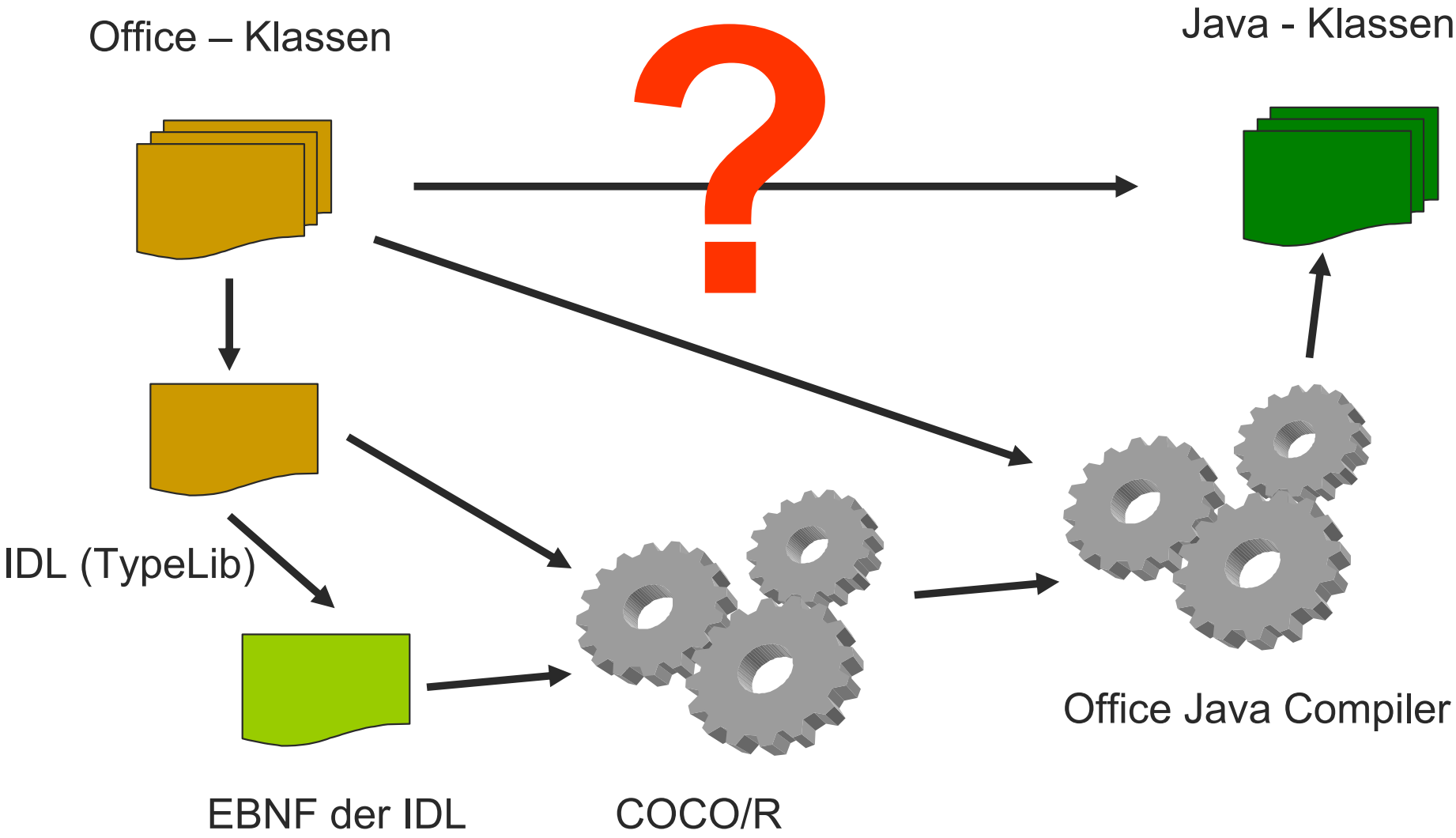
Ziel Delegationslayer

- Office verwenden wie wenn es Java wäre!!
  - Mühsame Typenkonvertierung soll weg fallen
  - Einfacher Aufruf von Methoden (nicht über mehrere Code Zeilen)
  - Javaobjekte sollen Office Objekte repräsentieren

Delegationlayer

- Erstellung sehr aufwendig um für jedes Office Objekt ein Java Objekt zu erstellen
- Ändert sich die Version des COM Objekts so muss der Delegationslayer angepasste werden
- Grosser Administrationsaufwand

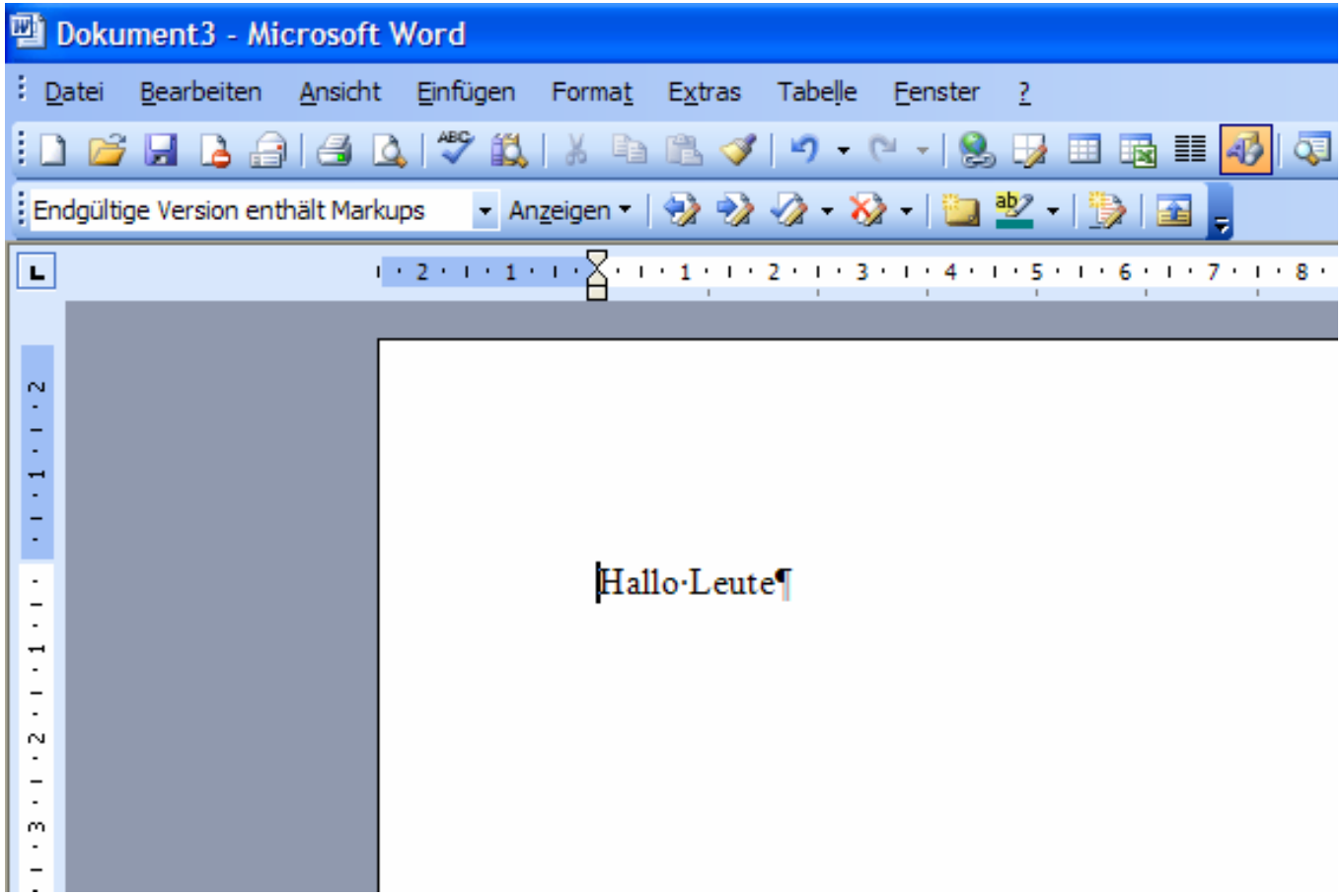
Delegationslayer stellen



„Hallo Leute“ mit Delegationslayer

```
Document document = Word.getWordDoc();
Application application = document.getApplication();
application.setVisible(true);

Documents docs = application.getDocuments();
Document newdoc = docs.Add();
Range range = newdoc.Range();
range.setText("Hallo Leute");
```

Resultat


Dokument3 - Microsoft Word
Datei Bearbeiten Ansicht Einfügen Format Extras Tabelle Fenster ?
Endgültige Version enthält Markups
Anzeigen
Hallo·Leute¶

Tipps COM in Java

- Erst in VB entwickeln → dann in Java
- Makro Recorder der Office Produkte verwenden
- Gute Newsgroups für Office - Programmierung

Referenzen

- http://jakarta.apache.org/poi/
- Lerhbuch der Software-Technik, Software-Entwicklung (Balzert) ISBN 3-8274-0480-0
- Visual Basic 6 Kompendium ISBN 3-8272-5806-5
- Eclipse ISBN 3-8273-2125-5
- COCO/R http://www.ssw.uni-linz.ac.at/Research/Projects/Coco/
- Office 2000 Developer Edition ISBN: 3-8272-5514-7