Advanced CouchDB

Slide extract for PHP Unconference Hamburg 2010

Kore Nordmann

September 27, 2010

Outline

Local conflict handling

- Implemented using “Multi-Version Concurrency Control” (MVCC)

```
1  $ $ curl -X GET http://localhost:5984/workshop_wiki/Start
2  { "_id":  "Start",
3    "_rev": "1-6bfd4885b6c62bb5169a19d5a81927e3",
4    "name": "Start",
5    "text": "Hello World!"
6  }
```

Local conflict handling

- Implemented using “Multi-Version Concurrency Control” (MVCC)

```
$ $ curl -X GET http://localhost:5984/workshop_wiki/Start
{ "_id":  "Start",
  "_rev": "1-6bfd4885b6c62bb5169a19d5a81927e3",
  "name": "Start",
  "text": "Hello World!"
}
```

- All documents in the database are versioned
- Updating the version is required for all operations modifying a document (update, delete)

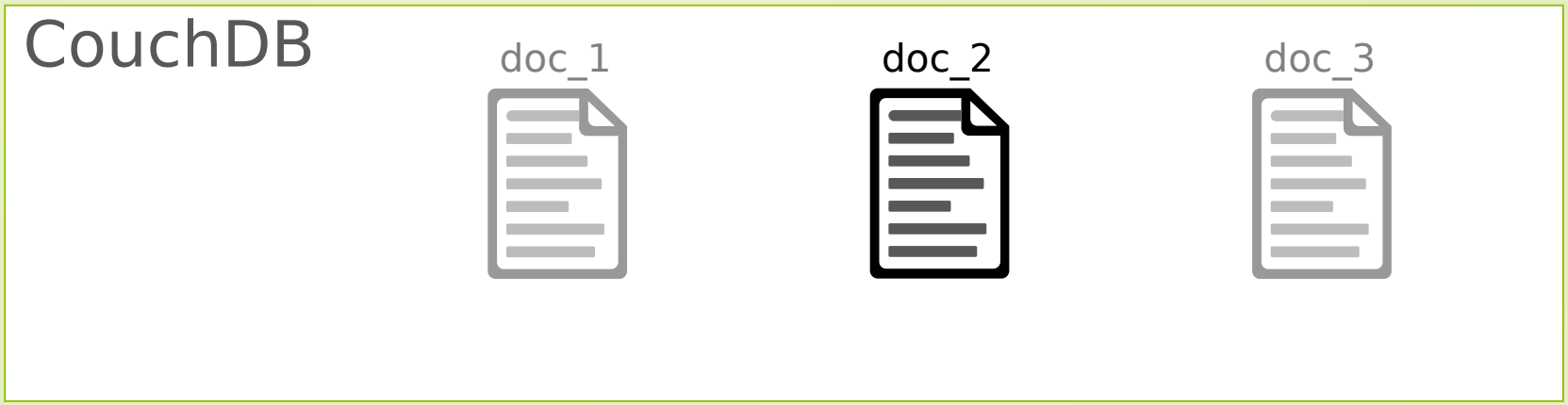
Local conflict handling

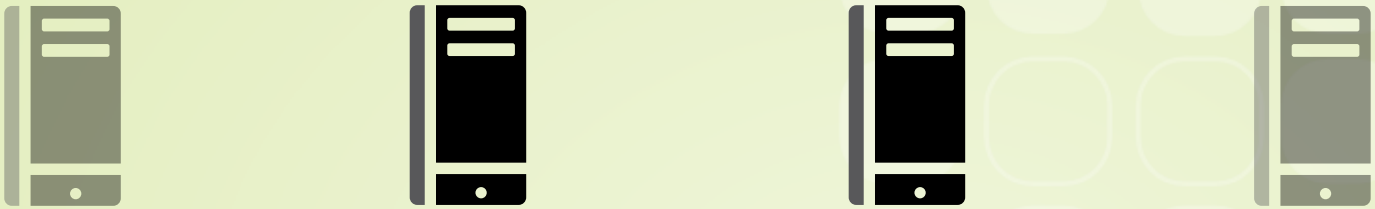
- Implemented using “Multi-Version Concurrency Control” (MVCC)

```
1 $ $ curl -X GET http://localhost:5984/workshop_wiki/Start
2 { "_id":  "Start",
3   "_rev": "1-6bfd4885b6c62bb5169a19d5a81927e3",
4   "name": "Start",
5   "text": "Hello World!"
6 }
```

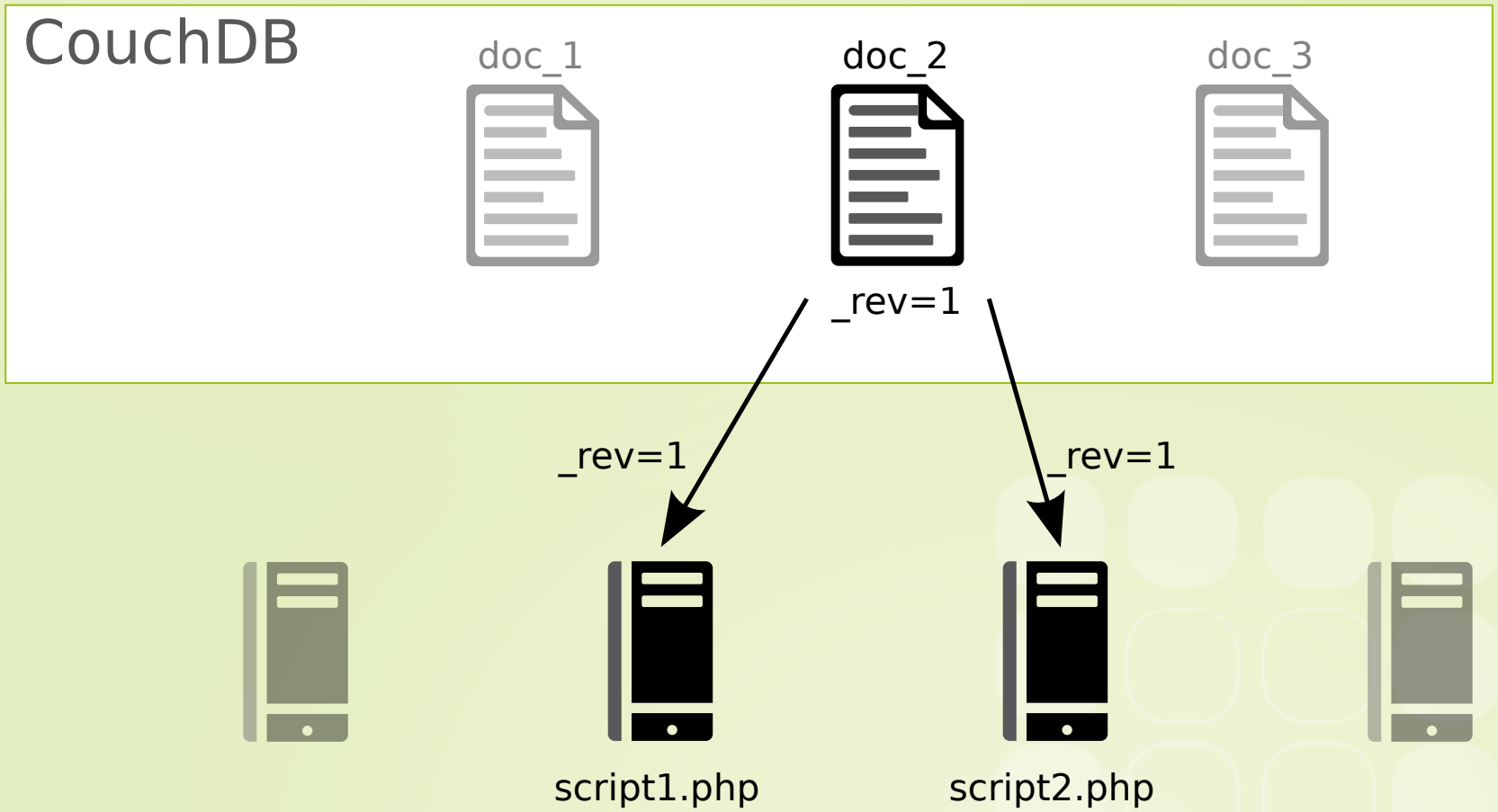
- All documents in the database are versioned
- Specifying the version is required for all operations mofifying a document (update, delete)

Local conflict handling

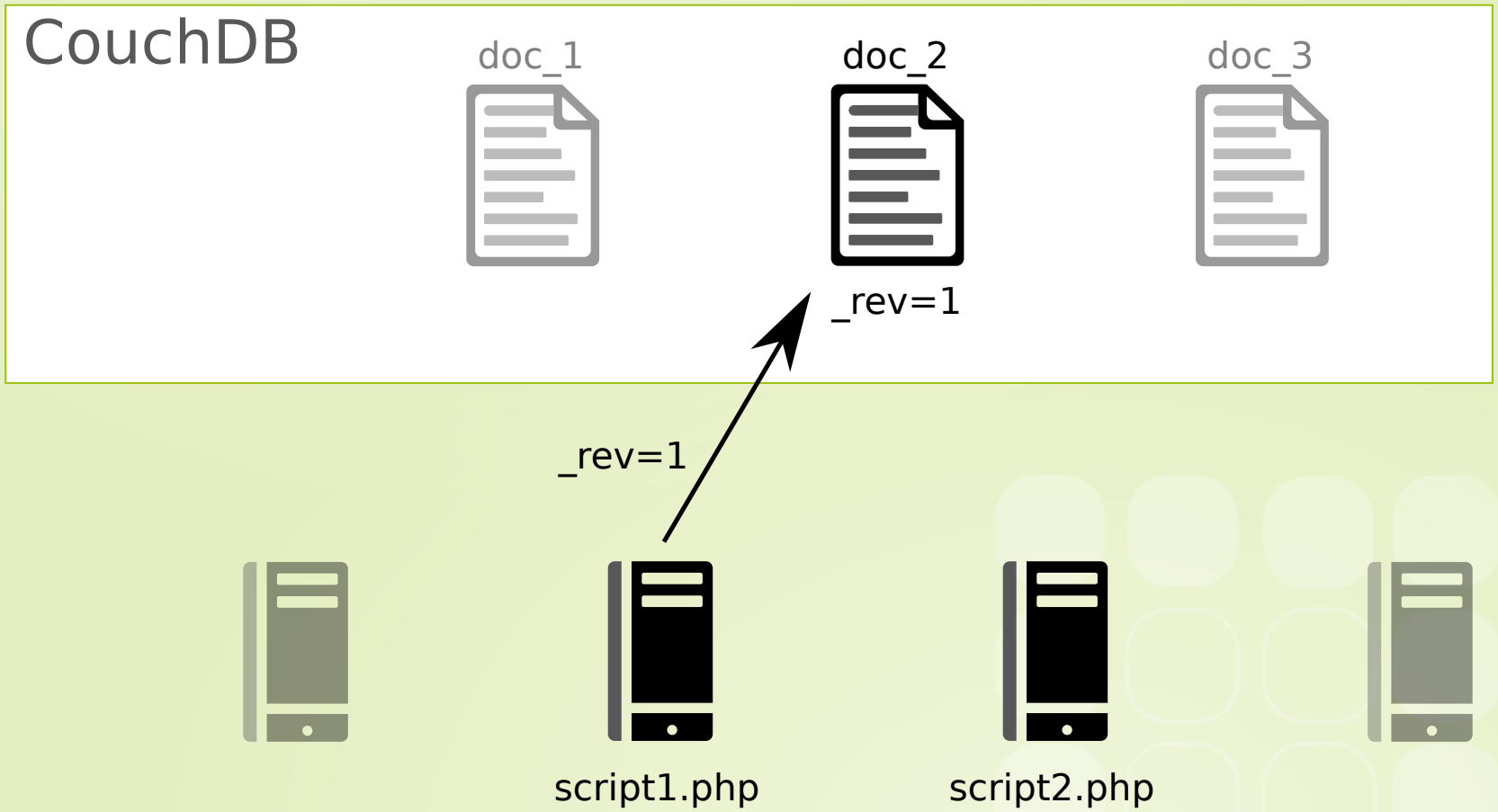




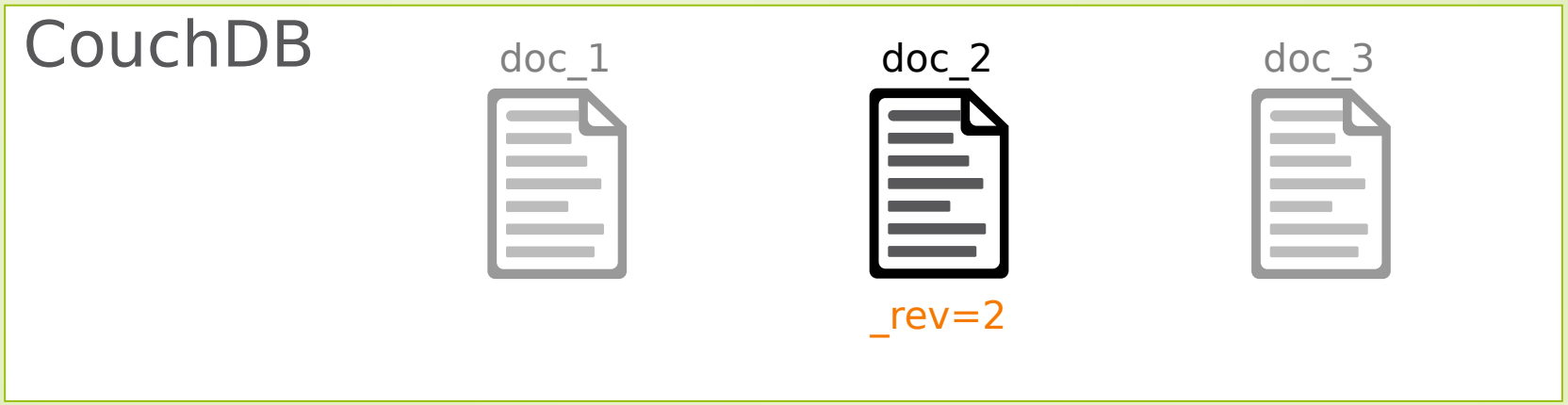
Local conflict handling

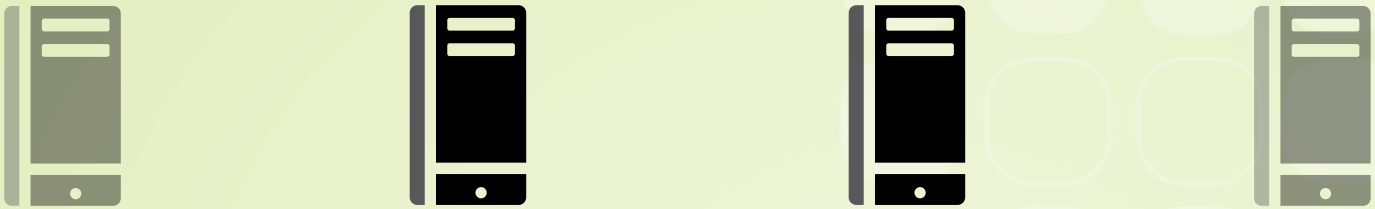


Local conflict handling

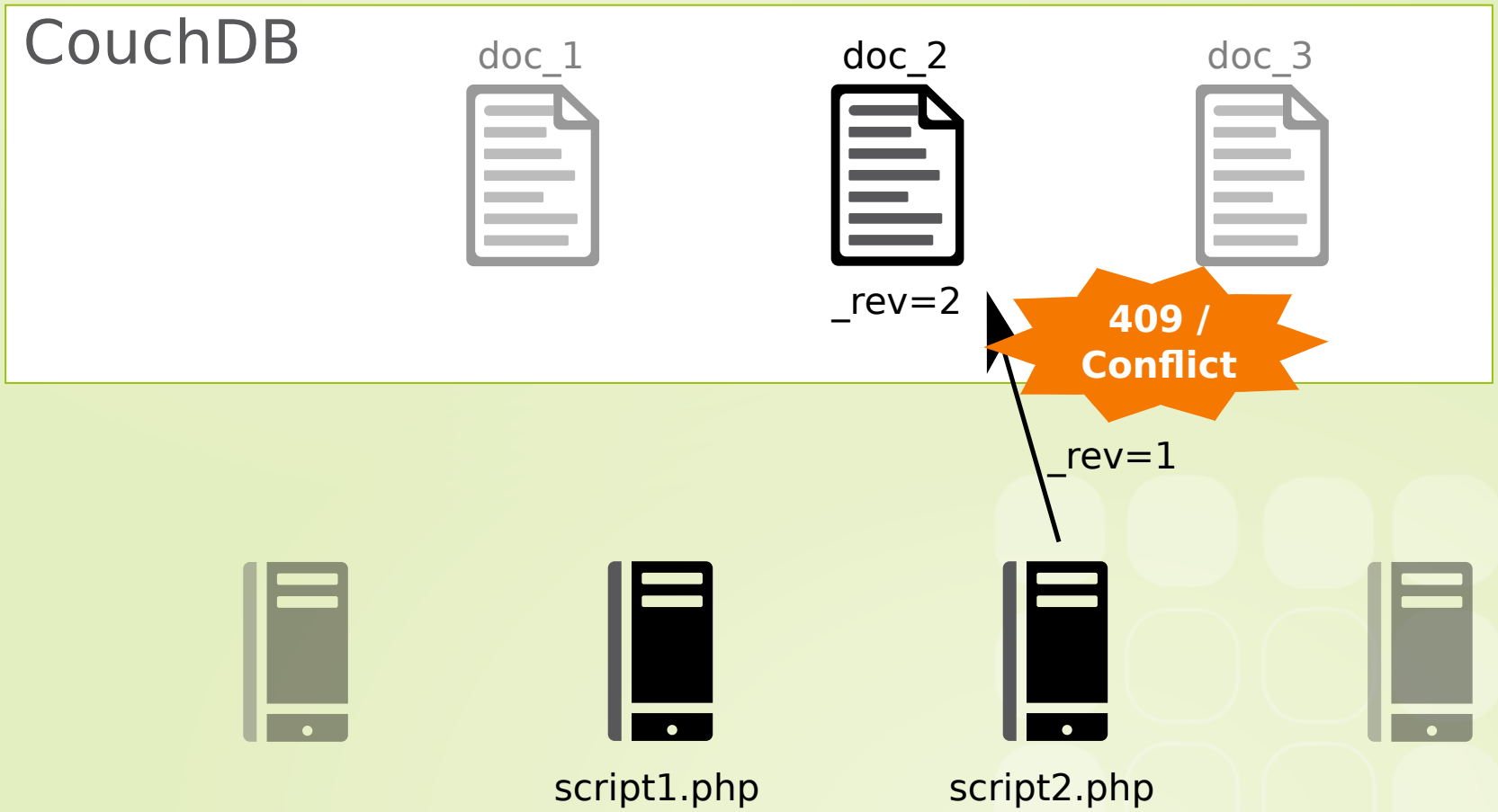


Local conflict handling





Local conflict handling



Local conflict handling

- Handling of conflicts is left to the application

Local conflict handling

- Handling of conflicts is left to the application
  - . . . there is no auto-magic way to merge two conflicting wiki documents anyways

Local conflict handling

- Handling of conflicts is left to the application
  - ... there is no auto-magic way to merge two conflicting wiki documents anyways
- You can also do this in RDBMS by adding a `revision` column
  - ... and probably should.

Pratical handling in CouchDB

- For document updates the revision property MUST be present in the JSON structure

```
1  $ curl -i -X PUT http://localhost:5984/workshop_wiki/Start --data '{"name": "
       Start", "text": "Hello World!"}'
2
3  HTTP/1.1 409 Conflict
4  Server: CouchDB/1.0.1 (Erlang OTP/R13B)
5  Date: Wed, 15 Sep 2010 10:43:22 GMT
6  Content-Type: text/plain;charset=utf-8
7  Content-Length: 58
8  Cache-Control: must-revalidate
9
10 {"error":"conflict","reason":"Document update conflict."}
```

Pratical handling in CouchDB

- For document updates the revision property MUST be present in the JSON structure

```
1  $ curl -i -X PUT http://localhost:5984/workshop_wiki/Start --data '{"name": "
       Start", "text": "Hello World!", "_rev": "1-6bfd4885b6c62bb5169a19d5a81927e3
       "}'
2
3  HTTP/1.1 201 Created
4  Server: CouchDB/1.0.1 (Erlang OTP/R13B)
5  Location: http://localhost:5984/workshop_wiki/Start
6  Etag: "2-cb54408a94198c03eb30d3eacb175951"
7  Date: Wed, 15 Sep 2010 10:44:15 GMT
8  Content-Type: text/plain;charset=utf-8
9  Content-Length: 68
10 Cache-Control: must-revalidate
11
12 {"ok":true,"id":"Start","rev":"2-cb54408a94198c03eb30d3eacb175951"}
```

Pratical handling in CouchDB

- For deletes specify the revision in the URL

```
1  $ curl -i -X DELETE http://localhost:5984/workshop_wiki/Start?rev=2-
       cb54408a94198c03eb30d3eacb175951
2
3  HTTP/1.1 200 OK
4  Server: CouchDB/1.0.1 (Erlang OTP/R13B)
5  Etag: "3-3d35f1daeb6457c1e925584f197d899e"
6  Date: Wed, 15 Sep 2010 10:45:09 GMT
7  Content-Type: text/plain;charset=utf-8
8  Content-Length: 68
9  Cache-Control: must-revalidate
10
11 {"ok":true,"id":"Start","rev":"3-3d35f1daeb6457c1e925584f197d899e"}
```

Pratical handling in CouchDB

- For deletes specify the revision in the URL

```
1  $ curl -i -X GET http://localhost:5984/workshop_wiki/Start
2
3  HTTP/1.1 404 Object Not Found
4  Server: CouchDB/1.0.1 (Erlang OTP/R13B)
5  Date: Wed, 15 Sep 2010 10:46:03 GMT
6  Content-Type: text/plain;charset=utf-8
7  Content-Length: 41
8  Cache-Control: must-revalidate
9
10 {"error":"not_found","reason":"deleted"}
```

Outline

Full-Text-Search

- Index all documents by all their words

```
1  function( doc ) {
2    if ( doc.type == "wiki" ) {
3      // Simple word indexing, does not respect overall occurences of words,
4      // stopwords, different word seperation characters, or word variations.
5      var text  = doc.title.replace( /[\s:.,!?-]+/g, " " ) +
6                  doc.text.replace( /[\s:.,!?-]+/g, " " );
7      var words = text.split( " " );
8      for ( var i = 0; i < words.length; ++i ) {
9        value = {};
10       value[doc._id] = 1;
11       emit( words[i].toLowerCase(), value );
12     }
13   }
14 }
```

Wiki

- Index all documents by all their words

```
...
"a"     => {wiki-8: 1}
"a"     => {wiki-8: 1}
"a"     => {wiki-8: 1}
"a"     => {wiki-8: 1}
"a"     => {wiki-81: 1}
"a"     => {wiki-83: 1}
"a"     => {wiki-83: 1}
"able"  => {wiki-39: 1}
"able"  => {wiki-56: 1}
"able"  => {wiki-73: 1}
"able"  => {wiki-80: 1}
"about" => {wiki-24: 1}
"about" => {wiki-43: 1}
"about" => {wiki-85: 1}
...
```

Full-Text-Search

- Reduce by word count

```
1  function( keys, values ) {
2    var count = {};
3    for ( var i in values ) {
4      for ( var id in values[i] ) {
5        if ( count[id] ) {
6          count[id] = values[i][id] + count[id];
7        } else {
8          count[id] = values[i][id];
9        }
10     }
11   }
12   return count;
13 }
```

Wiki

- Index all documents by all their words

```
...
"a"       => {
               wiki-68: 6,
               wiki-66: 6,
               wiki-22: 4,
               wiki-63: 3,
               wiki-60: 2,
               wiki-35: 2,
               wiki-34: 1,
               wiki-31: 1,
               ...
             }
"able"    => {wiki-86: 1, wiki-80: 1, wiki-73: 1, wiki-56: 1, wiki-39: 1}
"about"   => {wiki-85: 1, wiki-43: 1, wiki-24: 1}
...
```

Outline

Security

- By default CouchDB only listens on `127.0.0.1`
- By default everything is writeable and readable for everyone
  - Called “Admin-Party”

Security

- By default CouchDB only listens on `127.0.0.1`
- By default everything is writeable and readable for everyone
  - Called “Admin-Party”
- Once an administrator is created only the admin can create / remove databases

Security

- By default CouchDB only listens on `127.0.0.1`
- By default everything is writeable and readable for everyone
  - Called “Admin-Party”
- Once an administrator is created only the admin can create / remove databases
  - Can be created through Futon (try now)
  - Can be added to local.ini
  - Can be added by your application

Creating an admin user

```
$ cat admin.json
{
  "_id":           "org.couchdb.user:admin",
  "name":          "admin",
  "salt":          "68cf5946d9760d19759b5016d90f612c",
  "password_sha": "3588a9b2039e53b674d8da361e4be98f00637f5a",
  "type":          "user",
  "roles":         ["admin", "editor", "user"]
}

$ curl -i -X PUT http://127.0.0.1:5984/_users/org.couchdb.user%3Aadmin -d@admin.
    json

HTTP/1.1 201 Created
Server: CouchDB/1.0.1 (Erlang OTP/R13B)
Location: http://127.0.0.1:5984/_users/org.couchdb.user:admin
Etag: "1-cfcf3bbd31f67146180e23cf0fcc34af"
Date: Tue, 21 Sep 2010 15:02:54 GMT
Content-Type: text/plain;charset=utf-8
Content-Length: 85
Cache-Control: must-revalidate

{"ok":true,"id":"org.couchdb.user:admin","rev":"1-
    cfcf3bbd31f67146180e23cf0fcc34af"}
```

Authentification

- Handled by `/_session`
- Different auth mechanisms
  - HTTP Basic Auth
  - Cookie based auth
  - OAuth

```
1  $ curl -i -X POST http://127.0.0.1:5984/_session -d "name=kore&password=secret"
2
3  HTTP/1.1 200 OK
4  Set-Cookie: AuthSession=a29yZTo0Qzk4Q0IyQzpvdxKvB6ZHw2nhWgPLUYRM9Asm8g; Version
       =1; Path=/; HttpOnly
5  Server: CouchDB/1.0.1 (Erlang OTP/R13B)
6  Date: Tue, 21 Sep 2010 15:11:40 GMT
7  Content-Type: text/plain;charset=utf-8
8  Content-Length: 58
9  Cache-Control: must-revalidate
10
11 {"ok":true,"name":null,"roles":["_admin","admin","user"]}
```

Authentification

- Handled by `/_session`
- Different auth mechanisms
  - HTTP Basic Auth
  - Cookie based auth
  - OAuth
- POST your credentials to `/_session` as `application/x-www-form-urlencoded`

```
1  $ curl -i -X POST http://127.0.0.1:5984/_session -d "name=kore&password=secret"
2
3  HTTP/1.1 200 OK
4  Set-Cookie: AuthSession=a29yZTo0Qzk4Q0IyQzpvdxKvB6ZHw2nhWgPLUYRM9Asm8g; Version
       =1; Path=/; HttpOnly
5  Server: CouchDB/1.0.1 (Erlang OTP/R13B)
6  Date: Tue, 21 Sep 2010 15:11:40 GMT
7  Content-Type: text/plain;charset=utf-8
8  Content-Length: 58
9  Cache-Control: must-revalidate
10
11 {"ok":true,"name":null,"roles":["_admin","admin","user"]}
```

Authentification

- Handled by `/_session`
- Different auth mechanisms
  - HTTP Basic Auth
  - Cookie based auth
  - OAuth
- `POST` your credentials to `/_session` as `application/x-www-form-urlencoded`
- `GET /_session` verfies your login, if your client handles cookies.

```
1  $ curl -i -X POST http://127.0.0.1:5984/_session -d "name=kore&password=secret"
2
3  HTTP/1.1 200 OK
4  Set-Cookie: AuthSession=a29yZTo0Qzk4Q0IyQzpvdxKvB6ZHw2nhWgPLUYRM9Asm8g; Version
       =1; Path=/; HttpOnly
5  Server: CouchDB/1.0.1 (Erlang OTP/R13B)
6  Date: Tue, 21 Sep 2010 15:11:40 GMT
7  Content-Type: text/plain;charset=utf-8
8  Content-Length: 58
9  Cache-Control: must-revalidate
10
11 {"ok":true,"name":null,"roles":["_admin","admin","user"]}
```

Database access rights

- In databases everything is still writeable and readable for everyone
- You can specify users and groups which may read / write:
  - Once set, nobody else can access any more

```
$ curl -X GET http://admin:secret@127.0.0.1:5984/linuxhotel/_security

{ "admins": {
    "names": ["kore"],
    "roles": ["admin"]
  },
  "readers": {
    "names": [],
    "roles": ["user", "editor"]
  }
}
```

Database access rights

- ... but there is more fancy stuff:

```
1  function( newDoc, oldDoc, userContext ) {
2      if ( !userContext.name ) {
3          throw( { forbidden: "Please log in first." } );
4      }
5
6      if ( newDoc.username !== userContext.name ) {
7          throw( { unauthorized: "You may only edit your own documents." } );
8      }
9  }
```

- Add this to your design document under `validate_doc_update`

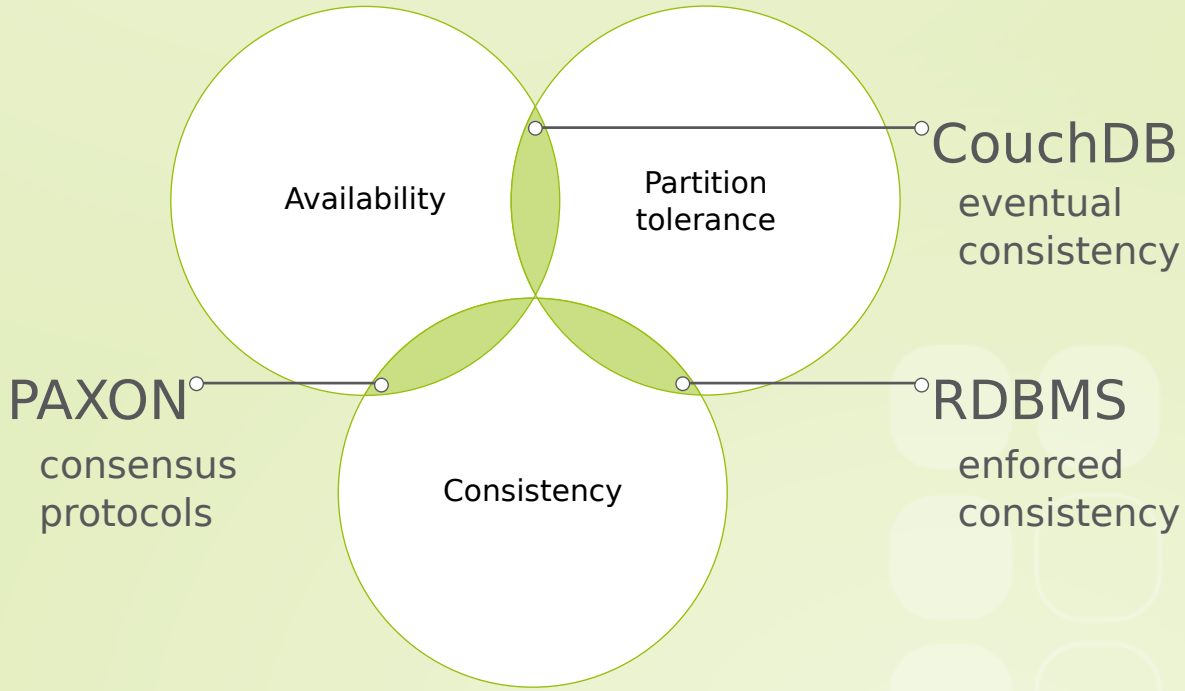
Outline

Remember:

- Multi-Version Concurrency Control (MVCC)
  - All documents in the database are versioned
- There is no ensured inter document consistency in CouchDB (relational integrity)

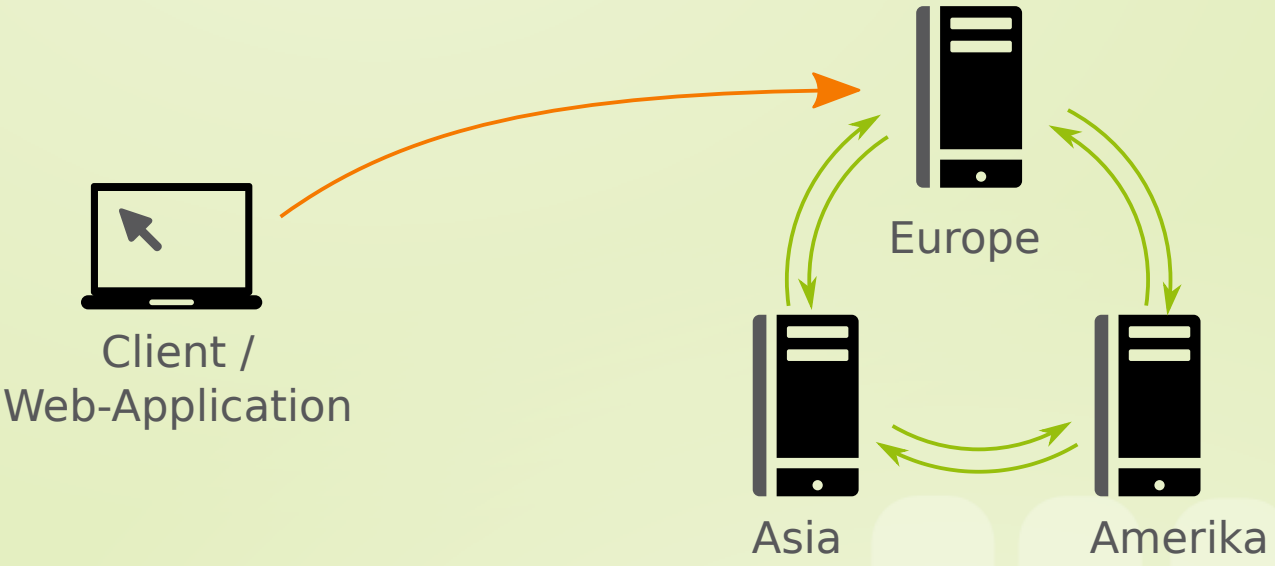
Scaling: The CAP theorem

- The CAP theorem, read more in “CouchDB: The Definitive Guide” [JCA09]



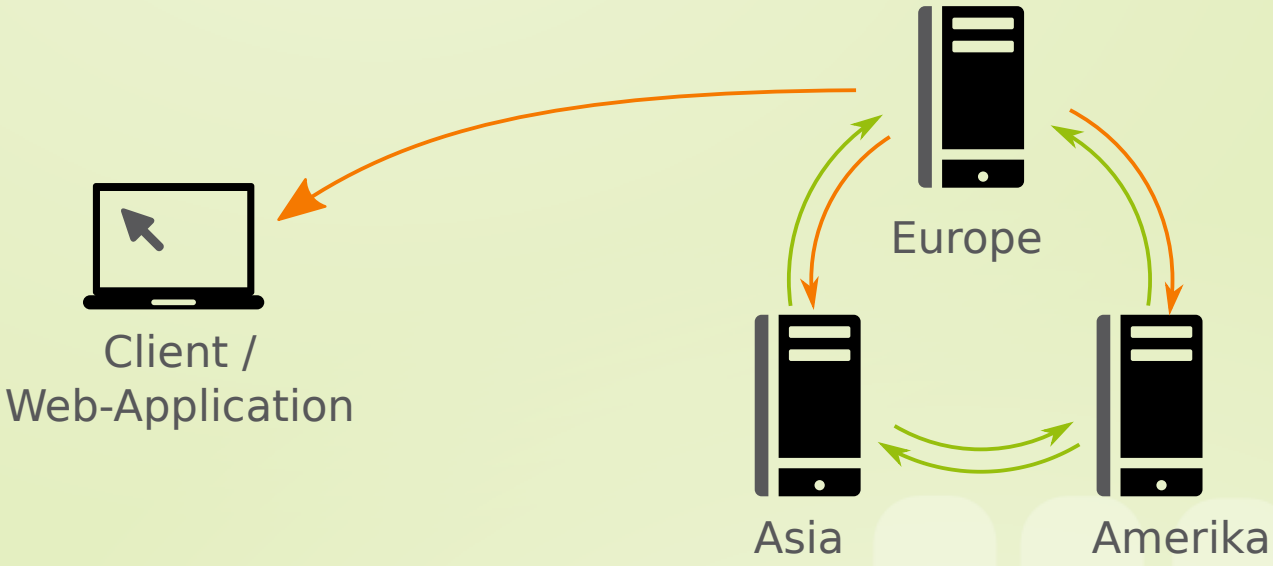
- CouchDB employs “Eventual Consistency” [Vog09]

Eventual consistency



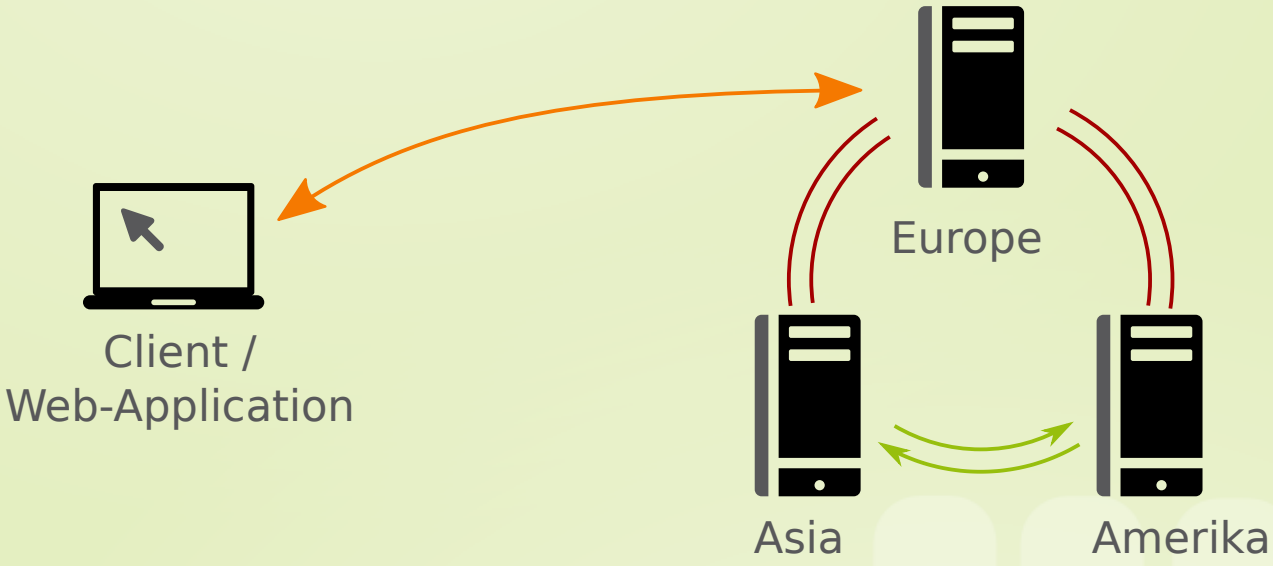
- Trigger triggered synchronization (push, pull)
- Deterministic (manual) conflict resolution on replication on all nodes
- Works well for seldom concurrent writes
- Structure your documents accordingly

Eventual consistency



- Delayed, triggered synchronization (push, pull)
  - Deterministic (manual) conflict resolution on replication on all nodes

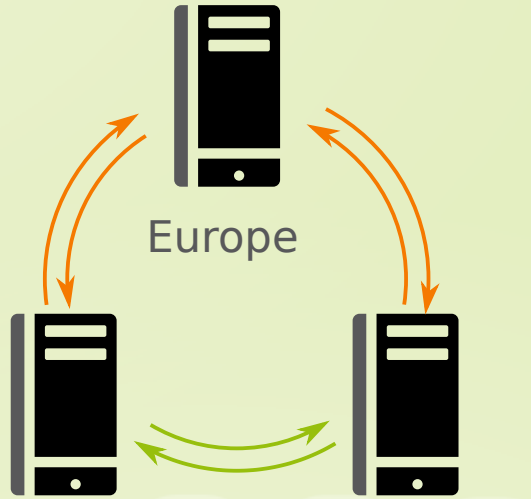
Eventual consistency



- Delayed, triggered synchronization (push, pull)
  - Deterministic (manual) conflict resolution on replication on all nodes

Eventual consistency

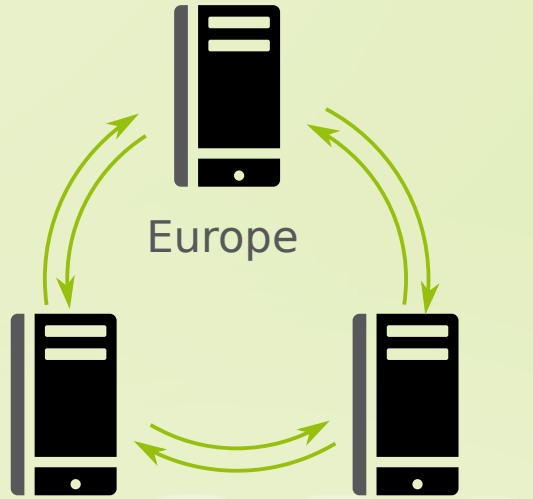
Client /
Web-Application



- Delayed, triggered synchronization (push, pull)
  - Deterministic (manual) conflict resolution on replication on all nodes
- Scales well for seldom concurrent writes
  - Structure your documents accordingly

Eventual consistency

Client /
Web-Application



- Delayed, triggered synchronization (push, pull)
  - Deterministic (manual) conflict resolution on replication on all nodes
- Scales well for seldom concurrent writes
  - Structure your documents accordingly

Outline

Replication

- Replication is trivial

```
$ curl -X POST http://localhost:5984/_replicate
    -H 'Content-Type: application/json'
    -d '{"source":"linuxhotel", "target":"http://user:pass@192.168.1.3:5984/
        linuxhotel"}'

{ "ok":                  true,
  "no_changes":          true,
  "session_id":          "73d69e7b5cdaea059e55ed1db7802151",
  "source_last_seq":     141,
  "history": [ {
    "session_id":            "73d69e7b5cdaea059e55ed1db7802151",
    "start_time":            "Thu, 23 Sep 2010 08:47:05 GMT",
    "end_time":              "Thu, 23 Sep 2010 08:51:53 GMT",
    "start_last_seq":        135,
    "end_last_seq":          141,
    "recorded_seq":          141,
    "missing_checked":       0,
    "missing_found":         1,
    "docs_read":             1,
    "docs_written":          1,
    "doc_write_failures":    0
  } ]
}
```

Replication details

- Source and target can be any combination of remote and local URLs

Replication details

- Source and target can be any combination of remote and local URLs
- Set {"continuous": true} to have CouchDB keeping the replication alive.

Replication details

- Source and target can be any combination of remote and local URLs
- Set {"continuous": true} to have CouchDB keeping the replication alive.
  - Is currently not persisted during restart

Replication details

- Source and target can be any combination of remote and local URLs
- Set {"continuous": true} to have CouchDB keeping the replication alive.
  - Is currently not persisted during restart
  - Can be cancelled again using {"cancel": true}

Replication details

- Source and target can be any combination of remote and local URLs
- Set {"continuous": true} to have CouchDB keeping the replication alive.
  - Is currently not persisted during restart
  - Can be cancelled again using {"cancel": true}
- Potential replication failures:
  - Crashed node
  - Network failure
  - validate_docs_update does not allow writing

Replication will be resumed once the error is fixed.

Replication details

- Source and target can be any combination of remote and local URLs
- Set {"continuous": true} to have CouchDB keeping the replication alive.
  - Is currently not persisted during restart
  - Can be cancelled again using {"cancel": true}
- Potential replication failures:
  - Crashed node
  - Network failure
  - validate_docs_update does not allow writing
    - Replication will be resumed once the error is fixed.

Checking replication status

```
1  $ curl -X GET http://user:pass@localhost:5984/_active_tasks
2
3  [ { "type":   "Replication",
4      "task":   "228689: http://koredn:*****@192.168.1.3:5984/linuxhotel/ ->
           linuxhotel",
5      "status": "W Processed source update #20",
6      "pid":    "<0.273.0>"
7    }, {
8      "type":   "Replication",
9      "task":   "0444e5: linuxhotel -> http://koredn:*****@192.168.1.3:5984/
           linuxhotel/",
10     "status": "MR Processed source update #141",
11     "pid":    "<0.292.0>"
12   }
13 ]
```

Filtered replication

- @TODO: Filtered replication

Thanks for listening

- More about us:
  - `http://qafoo.com`

Bibliography I

[JCA09] Noah Slater J. Chris Anderson, Jan Lehnardt, *Couchdb: The definitive guide*, O'Reilly Media, Inc., 2009.

[Vog09] Werner Vogels, *Eventually consistent - revisited*, `http://www.allthingsdistributed.com/2008/12/eventually_consistent.html`, December 2009.