



AD02: Overcoming Challenges in Collaborative Spreadsheet Editing with Shiny, SpreadJS and JSON-Patch

PHUSE EU Connect 2025, Hamburg

Kamil Foltyński, 17th November 2025



Agenda

Context & requirements

Architecture at a glance

Five challenges → solutions

Results & lessons

Q&A

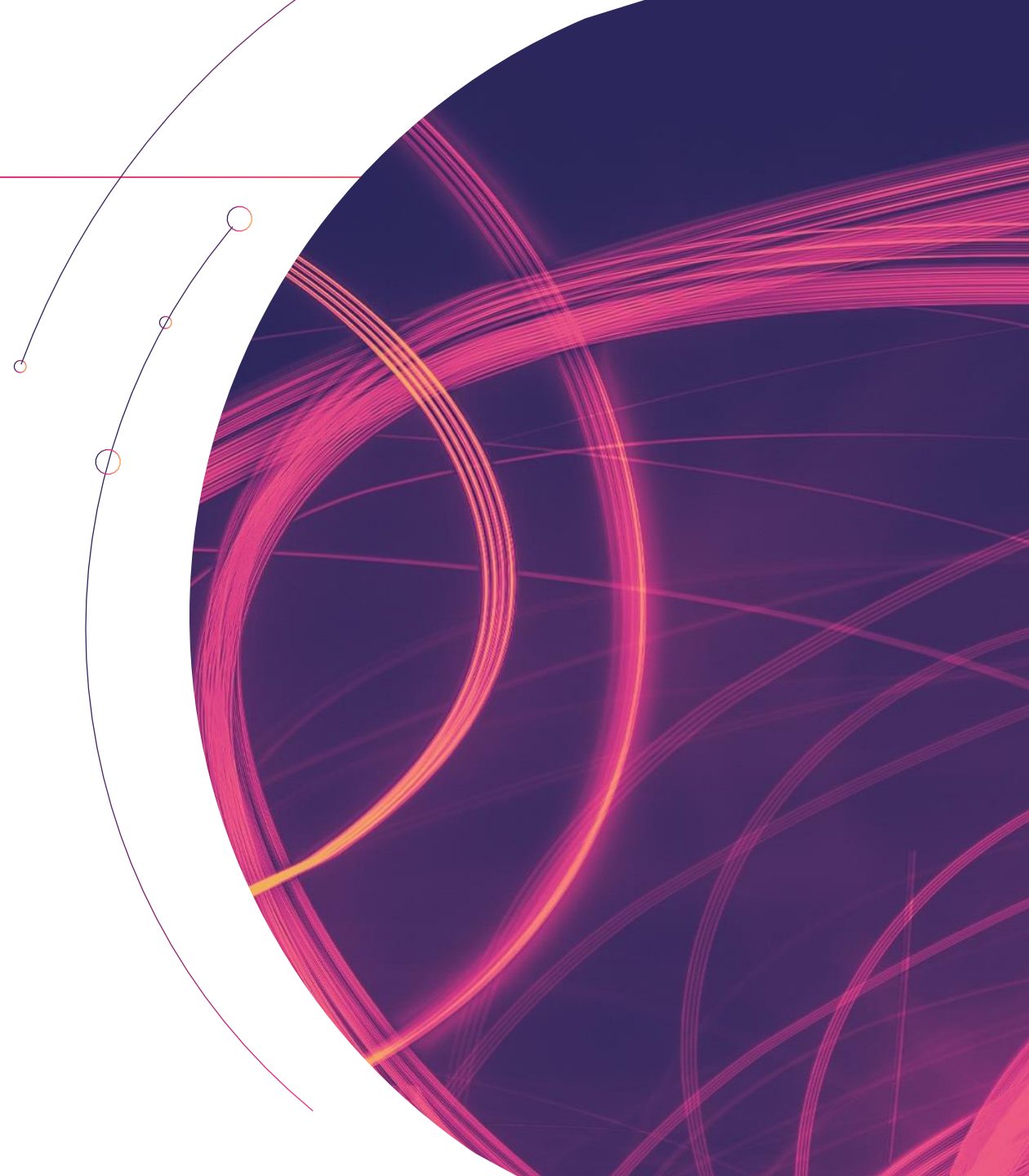
01

02

03

04

05

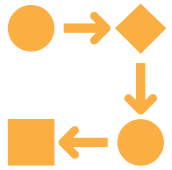


01

Context & requirements



Why not a generic cloud spreadsheet?



Cell-level audit trail
(who, what, when)



Server-side metadata
(author, last run, status)



Integrated dashboards
(custom reports)



Cross-checks (block
'Ready' if open issues)

Solution stack (why these three?)

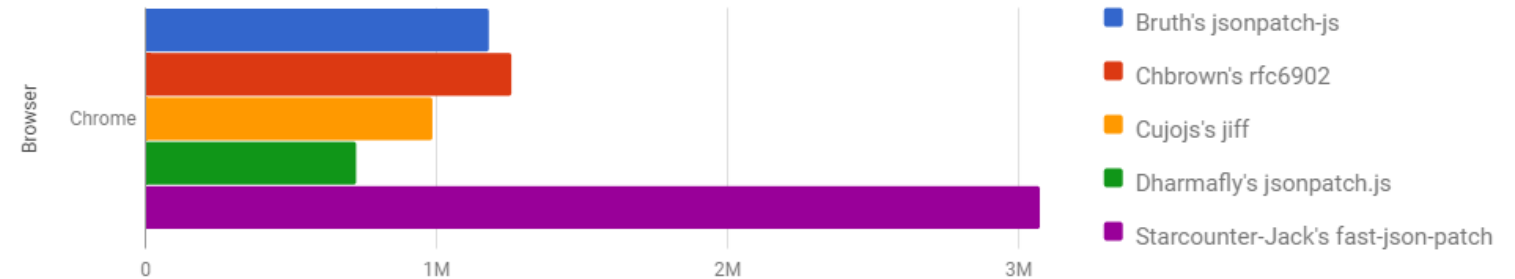
What is JSON-Patch?

JSON-Patch (**RFC6902**) is a standard format that allows you to update a JSON document by sending the changes rather than the whole document.

JSON-Patch implementation in **JavaScript** can perform millions of operations per second

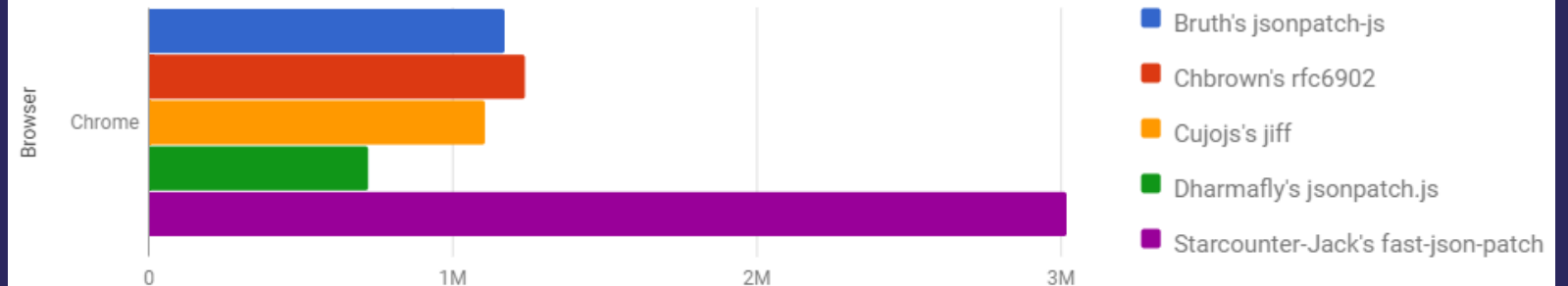
JSON Patch - Replace Operation

By operations/second – more is better – <https://perf.zone>



JSON Patch - Add Operation

By operations/second – more is better – <https://perf.zone>



02

Architecture at a glance



Architecture at a glance

SpreadJS Client
Shiny.setInputValue

Server

```
{ "op": "add", "path": "/sheets/Doc/data/dataTable/3",  
  "value": { "1": { "value": "change 1" } },  
  "user": "kamil", "timestamp": "2025-10-08T19:17:11.939700Z" },  
{ "op": "add", "path": "/sheets/Doc/data/dataTable/4",  
  "value": { "1": { "value": "change 2" } },  
  "user": "kamil", "timestamp": "2025-10-08T19:17:21.976400Z" },  
{ "op": "replace", "path": "/sheets/Doc/data/dataTable/4/1/value",  
  "value": "change 3",  
  "user": "sebastia", "timestamp": "2025-10-08T19:17:58.319800Z" },  
{ "op": "replace", "path": "/sheets/Doc/data/dataTable/4/1/value",  
  "value": "change 4",  
  "user": "kamil", "timestamp": "2025-10-08T19:17:50.204800Z" }
```

Audit & replay via **NDJSON** (changelog) + Git append

Prefer deltas; avoid full workbook round-trips

03

**Five challenges →
solutions**



Issues and Solutions



Excel-grade UX in a Shiny app

Excel-like behaviour and speed within Shiny's reactive UI.



Browser ↔ R integration

Reliable, low-latency two-way sync of grid events and data.



QA-acceptable audit trail

User-stamped, replayable, cell-level changes for compliance and review.



Megabyte-scale payloads

Replace full JSON with minimal JSON-Patch deltas to stay fast.



Concurrency & conflicts

Order and merge simultaneous edits deterministically using timestamps

1) Ex

```
// Compare baselines and snapshot, enrich with user/timestamp, debounce events
const jsonPatches = jsonpatch.compare(baseData, snapData).map((op, i) =>({...op,
  user: currentUser,
  timestamp: eventTimestamp,
  sequence: baseSeq + i
})));
Shiny.setInputValue(elementId + "_jsonpatches", {
  SpreadEvent: "exportJsonPatches",
  jsonStr: JSON.stringify(jsonPatches),
  args: args,
  user: currentUser,
  timestamp: eventTimestamp
},
{
  priority: "event"
});
// Event coalescing:
const handleSpreadEventDebounce = _.debounce(handleSpreadEvent, 50, {
  trailing: true,
  maxWait: 1500
});
eventsList.forEach(evt =>spread.bind(evt, handleSpreadEventDebounce));
```

ENTER

FORMAT

PUSH

DS

2) Integrating the grid with Shiny

Wrap

Wrap via `htmlwidgets`; wire two-way messaging

Emit

Emit events/data: `Shiny.setInputValue(...)`

Render

Render updates: `spreadJS(...)/renderSpreadJS(...)`

3) Building an audit trail

Export	Export full workbook JSON (SpreadJS.toJSON() method)
Append	Append JSON Lines to changelog (NDJSON)
Commit	Commit NDJSON with Git for immutable history
Replay	Replay patches to reconstruct any state

4) Heavy patch

```
Before: {"user": {"firstName": "Albert", "lastName": "Einstein"}}
After:  {"user": {"firstName": "Albert", "lastName": "Collins"}}
Patch:  [{"op": "replace", "path": "/user/lastName", "value": "Collins"}]
```



Client use

Add rows		
	A	B
1		
2		
3	Type	Description
4	Spec	specification description
5		
6		
6		

```
"contentStatus": ""
},
"docPropsApp": {
  "TotalTime": "0",
  "Application": "Microsoft Excel",
  "DocSecurity": "0",
  "ScaleCrop": "false",
  "HeadingPairs": {
    "size": 4,
    "baseType": "variant",
    "workbookInfoArray": [
      {
        "elementType": "vt:lpstr",
        "value": "Worksheets"
      },
      {
        "elementType": "vt:i4",
        "value": "4"
      }
    ]
  }
}
```



er": "kamil.fol
2399Z", "sequen

5) Concurrency & conflicts, timestamped ops

Misapply

Simultaneous edits
may be applied in
the wrong sequence

```
[
  {
    "op": "replace",
    "path": "/A1",
    "value": "Einstein",
    "user": "Kamil",
    "location": "Warsaw",
    "timestamp": "2025-11-12T14:23:45.123Z",
    "seq": 1
  },
  {
    "op": "replace",
    "path": "/A1",
    "value": "Collins",
    "user": "Sebastia",
    "location": "Hamburg",
    "timestamp": "2025-11-12T14:23:45.456Z",
    "seq": 2
  }
]
```

Replay

Replay
chronologically →
might final value for
the cell



04

Results & Lessons



Results & impact



Responsiveness: deltas → snappier UX



Auditability: restorable, user-stamped, cell-level trail (*NDJSON*)



Concurrency: timestamped + sequenced ops → **correct final value**



Integrated dashboards: in-app reviews, **no exports**, traceability kept

Lessons learned

01

Batch UI events
(~50 ms); throttle
sends

02

Patch order is
first-class concern
(timestamp +
sequence)

03

Stamp every
operation {user,
timestamp,
sequence}

04

Prefer deltas;
checkpoint full
JSON

05

Q&A



Q&A



QUESTIONS?



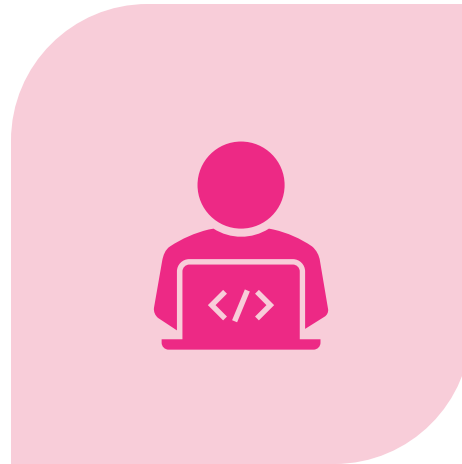
CONTACT:
KAMIL.FOLTYNSKI@CYTEL.COM

References

- IETF RFC 6902: JavaScript Object Notation (JSON) Patch - <https://www.rfc-editor.org/rfc/rfc6902>
- fast-json-patch (JavaScript/TypeScript implementation of JSON Patch) - <https://www.npmjs.com/package/fast-json-patch>
- SpreadJS (GrapeCity) - browser-native spreadsheet component - <https://www.grapecity.com/spreadjs>
- Shiny for R (Posit) - <https://shiny.posit.co/>
- htmlwidgets for R - <https://www.htmlwidgets.org/>

Contact

Your comments and questions are valued and encouraged. Contact author at:



KAMIL.FOLTYNSKI@CYTEL.COM

Thank you!

