A Context Sensitive Dashboard
For an Adaptive Knowledge Worker Assistant

Ralf Biedert, Sven Schwarz, Thomas Roth-Berghofer
Regarding Your Next Work-Day

- It is likely that you process several tasks a day
- Will interrupt your tasks, interrupt your interrupted tasks, ...
- Won’t finish all tasks a day
- Will do work twice

Thus, fragmentation of work is a way of life for these information workers. The majority of working spheres are interrupted and people spend about 11 minutes in a working sphere before switching to another. The irony of the work day is that the longer people spend in a working sphere, (and thus we assume become more involved in it) the more likely it is to be interrupted and the longer is the interrupting event“

From: No Task Left Behind! Examining the Nature of Fragmented Work. Mark et al.
Within MyMory Project at DFKI

- a context sensitive infrastructure and observation plugins were built
- MyDesk provides contexts
- plugins deliver native operations to the user observation hub
So, what *Exactly* is Context?

- **glbl. context (explicit)**
- a unique category provided by the user
- equals his selected task
- current local context (implicit)
- last observed NOPs
How do NOPs look like?

• „Native Operations“ for high-level user actions
• serialized as RDF
• examples: send mail, open document, navigate to, ...
• dispatched to listeners

```xml
<rdf:RDF
  xmlns:nop="http://ontologies.opendfki.de/repos/ontologies/..."
  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#">
  <nop:Application rdf:about="urn:app:Mozilla_Firefox">
    <nop:name>Mozilla Firefox</nop:name>
  </nop:Application>
  <nop:WebpageLoaded rdf:about="urn:nop:1212870387264_2">
    <nop:webUrl>
      <nop:Web rdf:about="http://de.start2.mozilla.com/"/>
    </nop:webUrl>
    <nop:certainty>1.0</nop:certainty>
    <nop:application rdf:resource="urn:app:Mozilla_Firefox"/>
  </nop:WebpageLoaded>
</rdf:RDF>
```
The Context Sensitive Dashboard

- stores users’ actions and contexts
- in addition monitors the screen and keyboard
- in order to
  - speed up users work *within* a context (principal idea: benefits must be higher than costs)
  - reduce mental context change times between context (provide retrieval cues)
CSD in Action
How do we Generate Assistance?

- The user is able to store bookmarks per context (always visible)
- most frequently used items (pC, pCS)
- similar situation (using CBR approach on a sliding window)
What is a *Similar Situation*?

- A *situation* is a window of observed events.
- Two windows are *similar* if they share similar events.
- This induces a ranking on all elements.

\[
\text{sim}(w_1, w_2) \rightarrow [0; 1]
\]
So what are Similar Events?

- depends on how pedantic you are. Please compare:

  open_page ap.wetter.com/forecast/germany/?sessionID=ZGF0ZT0yMDA4MD
  vs.

  print_page ap.wetter.com/forecast/germany/?sessionID=ZGF0ZT0yMDA4MD
  vs.

  open_page ap.wetter.com/forecast/germany/?sessionID=S5LCmKKL3yMVcIWP

- currently we are 'entity level pedantic'
Providing Excerpts

• every character is logged
• filter noise
• use top-1000 to filter non-important terms
• find terms with highest frequency
• extract area around
Evaluation Questions

• is there any benefit in employing (global) context to discriminate NOPs and entities?

• does everyone benefit?

• what widgets are useful, what elements should they contain?

• are the costs worth the benefits?

• does the take-me-back help you to remember what you did (last summer)?
Is There a Benefit of a g. Context?

- does it make sense to have a global context?
- wouldn’t it be easier to have one giant „bucket“?
- careful: don‘t replace entity access problem with context access problem!
Does Everyone Benefit?

• is a hard context switch a „natural“ way of interaction?

• „entity centered context definition“ vs. „action centered context definition“?

• mixed „pierce through“?
Elements to Select?

- obvious: listen to users‘ recommendations about new widgets
- to find out which widgets are used frequently and which are ignored most of the time
- monitor which category of provided elements were used most frequently (favorites, frequency, situation) but take into account the display issues (favorite superseded frequency superseded situation)
Do Benefits Outweigh Costs?

• very tough question
• easy part: costs inside dashboard
• difficult part: calculate 'alternative' costs
• frankly, no idea how to do this practically. In theory just watch every step
So, what will we do?

• offline evaluation of similar-situation algorithm with leave-one-out-cross-validation

• user studies with a number of students „please use it for an hour and tell us what you think“

• *Fragmented Task Coping Test*: one hour, every five minutes (times might change) task will be changed, thus 12 fragments. Pretest: no support. Test: dashboard. Posttest: no support. Compare task throughput.
Thank You!