Posrednici umreženih sustava

Prof.dr.sc. Siniša Srbljić

Dr.sc. Ivan Benc

Dr.sc. Daniel Skrobo

Fakultet elektrotehnike i računarstva
Laboratorij za potrošaču prilagođeno računarstvo

10. Predavanje

Formalno modeliranje raspodijeljenih sustava zasnovanih na posrednicima; SIP/WS posrednik

Ivan Budiselić, dipl.ing.

Fakultet elektrotehnike i računarstva
Laboratorij za potrošaču prilagođeno računarstvo

Sadržaj predavanja

- **Formalno modeliranje raspodijeljenih sustava zasnovanih na posrednicima**
  - Uvod u formalno modeliranje RSZNP
  - Architectural Interaction Diagrams
- **SIP/WS posrednik**
  - SIP i SOAP protokoli
  - Primjer: pristup mreži osjetila protokolom uspostave sjednice
  - Arhitektura SIP/WS posrednika
  - Primjena SIP/WS posrednika za pristup mreži osjetila primjenom protokola uspostave sjednice
- **Literatura**

Formalno modeliranje RSZNP

- **Zašto se računalni sustavi modeliraju?**
  - Rano otkrivanje grešaka u dizajnu
  - Formalna verifikacija važnih svojstava sustava
- **Modeliranje sklopovlja vs. modeliranje programske potpore**
  - Gotovo neizbježno se koristi kod dizajna sklopovlja
  - Modeliranje programske potpore: područje intenzivnog istraživanja
- **Alternativa modeliranju programske potpore (praksa)**
  - Testiranje sustava na svim razinama (*unit testing*)
  - Povezivanje elemenata sustava (*integration testing*)
  - Problem kod RSZNP?
    - Nedeterminizam i složeno međudjelovanje elemenata

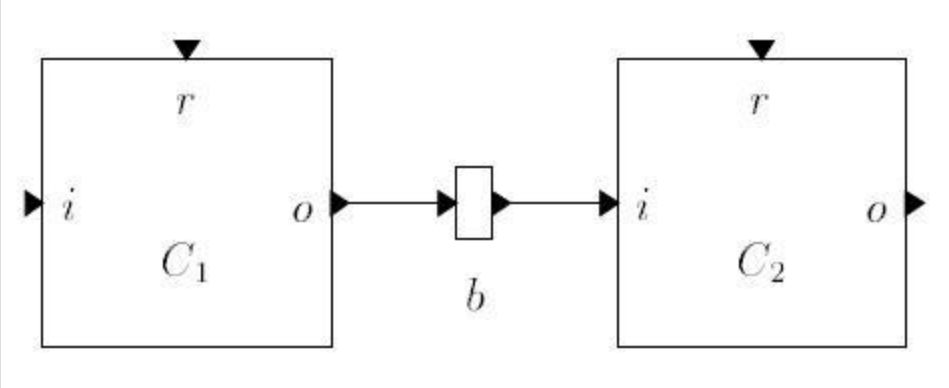
Formalno modeliranje RSZNP

- **Formalni sustavi zasnovani na strojevima stanja**
  - Konačni automati
  - Statecharts (podržava modeliranje paralelnog izvođenja)
  - Message Sequence Charts (protokoli između čvorova sustava)
  - Petrijeve mreže
- **Ostali formalni sustavi**
  - Procesna algebra (npr. Communicating Sequential Processes, π-calculus)
  - Teorija redova

Architectural Interaction Diagrams

- **Naglasak na međudjelovanju u širem smislu**
  - Komunikacija između procesa
  - Npr. modeliranje pouzdanog/nepouzdanog kanala, sinkrona/asinkrona komunikacija…
- **Pogodno za modeliranje RSZNP**
  - Za RSZNP karakteristična je intenzivna interakcija među čvorovima uz korištenje raznih sinkronizacijskih mehanizama

Architectural Interaction Diagrams



- **Osnovni elementi AID formalnog sustava**
  - I/O Labeled Transition System (IOLTS)
  - Sabirnice
  - Veze između pristupnih točaka
    - Smjerovi strelica određuju radi li se o ulaznoj ili izlaznoj pristupnoj točki
    - Najviše jedna veza ulazi ili izlazi iz određene pristupne točke

Architectural Interaction Diagrams

- **I/O Labeled Transition System**
  - Modeliranje čvorova sustava
  - Svaki IOLTS može interno biti opisan bilo kojim formalnim modelom, dok god se sučelje i ponašanje mogu prevesti u sljedeći oblik
  - Formalna definicija:

$$(I, Q, T, q_0)$$

I – sučelje; poredani par skupova ulaznih i izlaznih pristupnih točaka

Q – skup stanja, $q_0 \in Q$ početno stanje

T – skup prijelaza

Architectural Interaction Diagrams

- **Prijelazi IOLTS elemenata**
  - promjena stanja pisanjem - (q, t_w ! v, q')
    - q – početno stanje; q' – konačno stanje
    - t_w - izlazna pristupna točka; v – vrijednost
  - promjena stanja čitanjem - (q, t_r ?, q')
    - t_r – ulazna pristupna točka
  - promjena stanja bez međudjelovanja s okolinom - (q, q')
  - promjena stanja uz atomičko obavljanje slijeda operacija čitanja i pisanja
    - npr. (q, t_w!v; t_r?, q') označava prelazak iz stanja q u stanje q' uz zapisivanje vrijednosti v na izlaznu pristupnu točku t_w nakon čega neposredno slijedi čitanje podatka sa ulazne pristupne točke t_r

Architectural Interaction Diagrams

- **Sabirnice**
  - Modeliranje međudjelovanja elemenata sustava – “pokreću” model
  - Formalna definicija:

$$(I, B, T, b_0)$$

I – sučelje; poredani par skupova ulaznih i izlaznih pristupnih točaka

B – skup stanja, $b_0 \in B$ početno stanje

T – skup prijelaza

Architectural Interaction Diagrams

- **Prijelazi sabirnice**

b – početno stanje; **b'** – konačno stanje

W – skup ulaznih pristupnih točaka

R – skup izlaznih pristupnih točaka

WV – skup poredanih parova (ulazna pristpuna točka, vrijednost)

RV – skup poredanih parova (izlazna pristpuna točka, vrijednost)

Architectural Interaction Diagrams

- **Prijelazi sabirnice**

$$\mathbf{b} \xrightarrow[WV\ R]{W\ RV} \mathbf{b'}$$

  - Ako komponenta modela spojena preko izlazne pristupne točke **w'** na ulaznu pristupnu točku **w** sabirnice u trenutnom stanju **p** ima prijelaz oblika **(p, w' ! v, p')**, onda je **(w, v)** $\in$ **WV**
  - Ako komponenta modela spojena preko ulazne pristupne točke **r'** na izlaznu pristupnu točku **r** sabirnice u trenutnom stanju **p** ima prijelaz oblika **(p, r' ?, p')**, onda je **r** $\in$ **R**

Architectural Interaction Diagrams

- **Prijelazi sabirnice**

  - Uz prijelaz se definira predikat **P**(**WV**, **R**)
    - Ako je **P** istinit, sabirnica prelazi is stanja **b** u stanje **b'** i u spojenim sustavima aktivira prijelaze za pisanje na pristupne točke u **W(WV, R)** i čitanje u skladu s **RV(WV, R)** (**W** i **RV** također ovise o **WV** i **R**, tj. o početnim uvjetima)

Architectural Interaction Diagrams

- **Primjer: sinkrona razmjena poruka**
  - B = { b } (sabirnica modelira međudjelovanje u kojem se ne čuva stanje)
  - T sadrži prijelaz b $\xrightarrow[WV\ R]{W\ RV}$ b

P(**WV**, **R**) = $\exists(\mathbf{t_w}, \mathbf{x}) \in \mathbf{WV} \wedge \mathbf{t_r} \in \mathbf{R}$

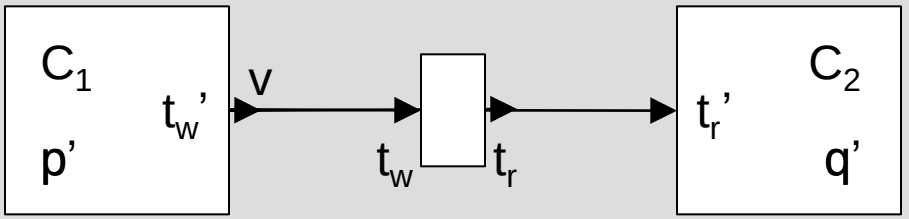
W(**WV**, **R**) = { $\mathbf{t_w}$ }

RV(**WV**, **R**) = { ($\mathbf{t_r}$, **x**) | ($\mathbf{t_w}$, **x**) $\in$ **WV** }

$P(\{(t_w, v)\}, \{t_r\}) = true$

$W(\{(t_w, v)\}, \{t_r\}) = \{ t_w \}$

$RV(\{(t_w, v)\}, \{t_r\}) = \{ (t_r, v) \}$



$(p, t_w' ! v, p') \in T_1$

$WV = \{ (t_w, v) \}$

$(q, t_r' ?, q') \in T_2$

$R = \{ t_r \}$

Architectural Interaction Diagrams

- **Primjer: *broadcast***
  - B = { b }
  - T sadrži prijelaz $\mathbf{b} \xrightarrow[WV\ R]{W\ RV} \mathbf{b}$

$\mathbf{P(WV, R)} = \exists(\mathbf{t_w}, \mathbf{x}) \in \mathbf{WV} \wedge \exists \mathbf{t_r} \in \mathbf{R}$

$\mathbf{W(WV, R)} = \{ \mathbf{t_w} \}$

$\mathbf{RV(WV, R)} = \{ (\mathbf{t_r}, \mathbf{x}) \mid \mathbf{t_r} \in \mathbf{R}, (\mathbf{t_w}, \mathbf{x}) \in \mathbf{WV} \}$

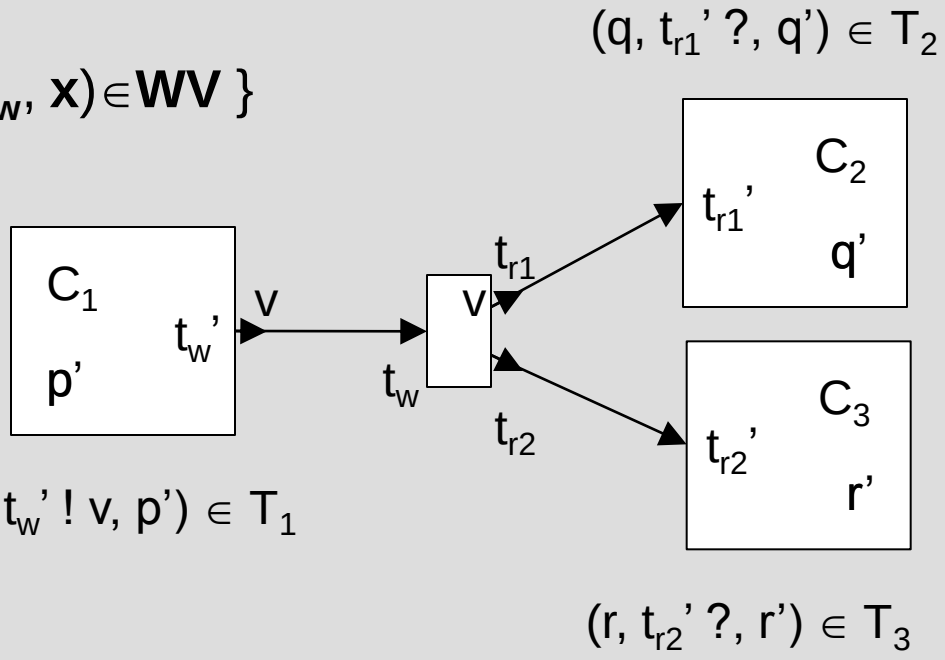
$WV = \{ (t_w, v) \}$ $R = \{t_{r1}, t_{r2}\}$

$P(\{(t_w, v)\}, \{t_{r1}, t_{r2}\}) = true$

$W(\{(t_w, v)\}, \{t_{r1}, t_{r2}\}) = \{ t_w \}$

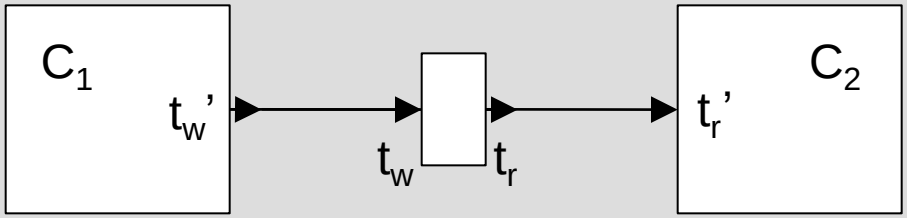
$RV(\{(t_w, v)\}, \{t_{r1}, t_{r2}\}) =$

$\{ (t_{r1}, v), (t_{r2}, v) \}$



Architectural Interaction Diagrams

- **Primjer: spremnik događaja (event heap, event channel...)**
  - Interakcijski posrednik koji pohranjuje *događaje* (npr. XML datoteke) i *pretplate* na događaje
  - Svaka pretplata je poredani par (maska, izlazna pristupna točka) i definira da se pristigli događaji koji *odgovaraju* zadanoj maski šalju na odgovarajuću pristupnu točku
  - Za pristupnu točku *t* sabirnice, definiramo *komp(t)* kao jedinstveni identifikator komponente koja je spojena na sabirnicu kroz *t*



$komp(t_w) = C_1$, $komp(t_r) = C_2$

Architectural Interaction Diagrams

- **Primjer: spremnik događaja (event heap, event channel...)**
  - Događaji i pretplate pohranjuju se u spremniku u listama L_D i L_P
  - *Stanje* sabirnice koja modelira spremnik definirano je sadržajem L_D i L_P (u početnom stanju su obje liste prazne)

Architectural Interaction Diagrams

- **Primjer: spremnik događaja (event heap, event channel...)**
  - Prijelazi za neblokirajući dohvat podataka (poruka *p* sadrži masku):
  - Stanje sabirnice se ne mijenja

$\mathbf{P}(\mathbf{WV}, \mathbf{R}) = \exists(\mathbf{t_w}, \mathbf{x}) \in \mathbf{WV} \wedge \exists \mathbf{t_r} \in \mathbf{R} \wedge komp(t_w) = komp(t_r)$

$\mathbf{W}(\mathbf{WV}, \mathbf{R}) = \{ \mathbf{t_w} \}$

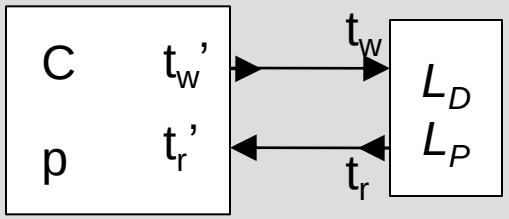
$\mathbf{RV}(\mathbf{WV}, \mathbf{R}) = \{ (\mathbf{t_r}, \mathbf{d}) \mid (\mathbf{t_w}, \mathbf{x}) \in \mathbf{WV}, \mathbf{d} \in Maskiraj(\mathbf{x}) \}$

pri čemu je *Maskiraj*(**x**) skup svih događaja koji odgovaraju maski pohranjenoj u poruci **x**

$P(\{(t_w, p)\}, \{t_r\}) = true$

$W(\{(t_w, p)\}, \{t_r\}) = \{ t_w \}$

$RV(\{(t_w, p)\}, \{t_r\}) = \{ (t_r, d) \mid d \text{ po definicij} \}$



$(p, t_w'!p; t_r'?, p') \in T_c$

Sadržaj predavanja

- **Formalno modeliranje raspodijeljenih sustava zasnovanih na posrednicima**
  - Uvod u formalno modeliranje RSZNP
  - Architectural Interaction Diagrams
- **SIP/WS posrednik**
  - SIP i SOAP protokoli
  - Primjer: pristup mreži osjetila protokolom uspostave sjednice
  - Arhitektura SIP/WS posrednika
  - Primjena SIP/WS posrednika za pristup mreži osjetila primjenom protokola uspostave sjednice
- **Literatura**

Protokol uspostave sjednice (SIP)

- **Funkcije protokola uspostave sjednice:**
  - Pronalazak korisnika (*user location*)
  - Ispitivanje raspoloživosti korisnika (*user availability* / *presence*)
  - Ispitivanje mogućnosti korisnika (*user capabilities*)
  - Postavljanje sjednice (*session setup*)
  - Upravljanje sjednicom (*session managemet*)
- **Primjena**
  - Pokretni uređaji
  - VoIP, video
  - Brojna proširenja protokola (IM, dojava događaja)

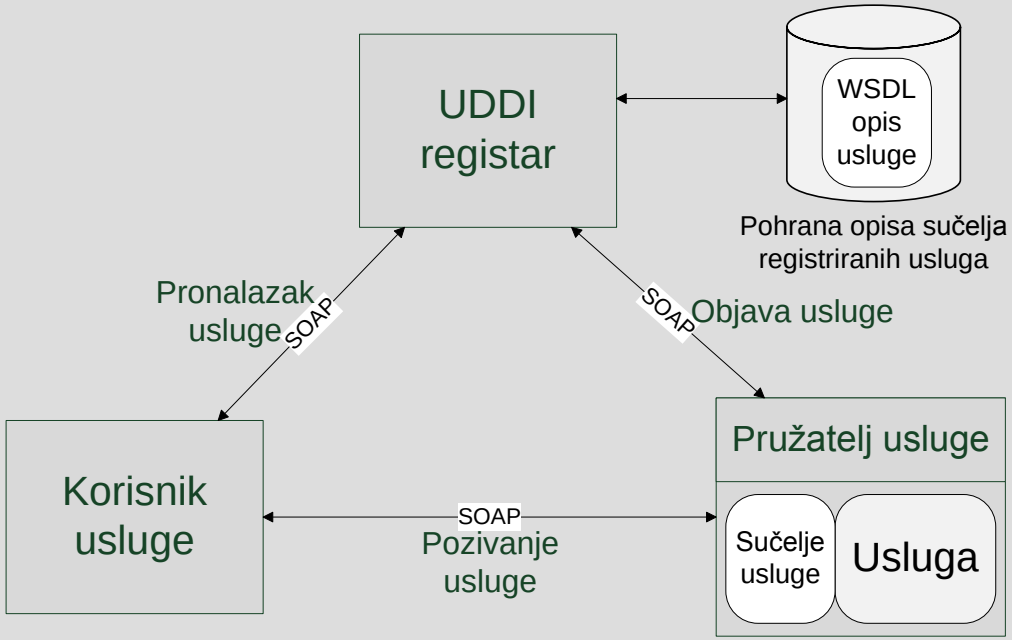
Protokol uspostave sjednice (SIP)

- **Znakovni protokol (oblikom sličan HTTPu)**

```
INVITE sip:korisnik2@domena2.com SIP/2.0
Via: SIP/2.0/UDP sip.domena1.com;branch=z9hG4bK76mersda
Max-Forwards: 60
To: Korisnik2 <sip:korisnik2@domena2.com>
From: Korisnik1 <sip:korisnik1@domena1.com>;tag=2447852776
Call-ID: c87b4676e66az0@sip.domena1.com
CSeq: 315013 INVITE
Contact: <sip:korisnik1@sip.domena1.com>
Content-Type: application/sdp
Content-Length: 133
```

- **Često se koristi kao prijenosni protokol za poruke drugih protokola (npr. Session Description Protocol)**

SOAP protokol



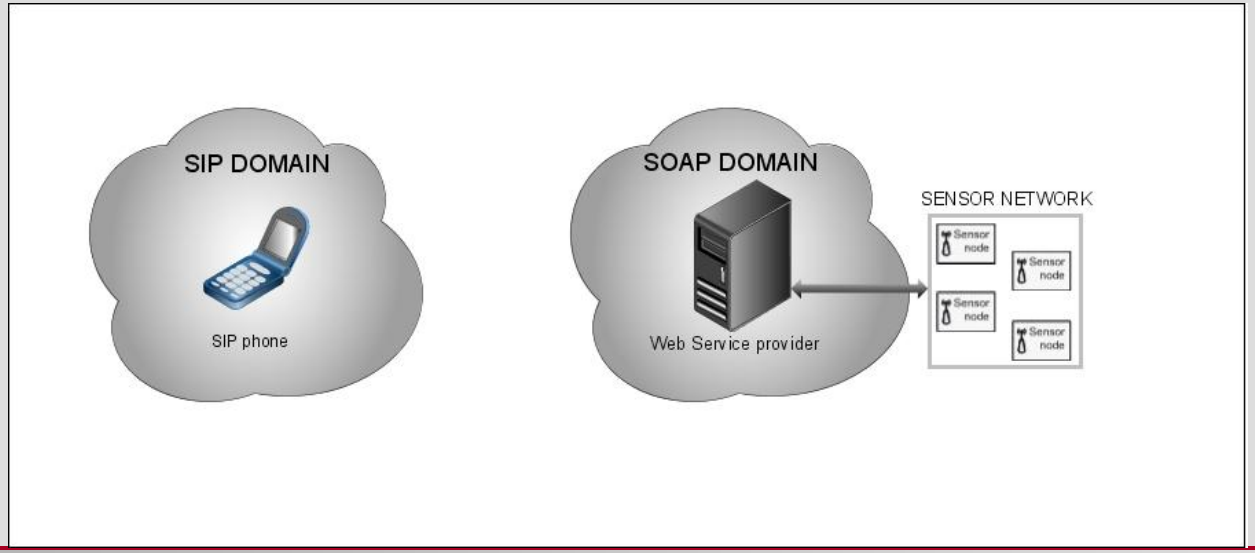
- **Protokol za razmjenu poruka u okviru Web Services tehnologije**
- **Koristi se najviše u poslovnim sustavima**
  - Pogodno za povezivanje sa sustavima poslovnih partnera
  - Neovisnost o platformi
- **Zahtjev-odgovor protokol (*request response*)**

Motivacija

- **Zašto povezivati SIP i SOAP protokole?**
  - Značajan broj postojećih SIP ili SOAP usluga (nova funkcionalnost za korisnike)
  - Mnogi pokretni uređaji ne podržavaju SOAP protokol (npr. SIP telefoni)

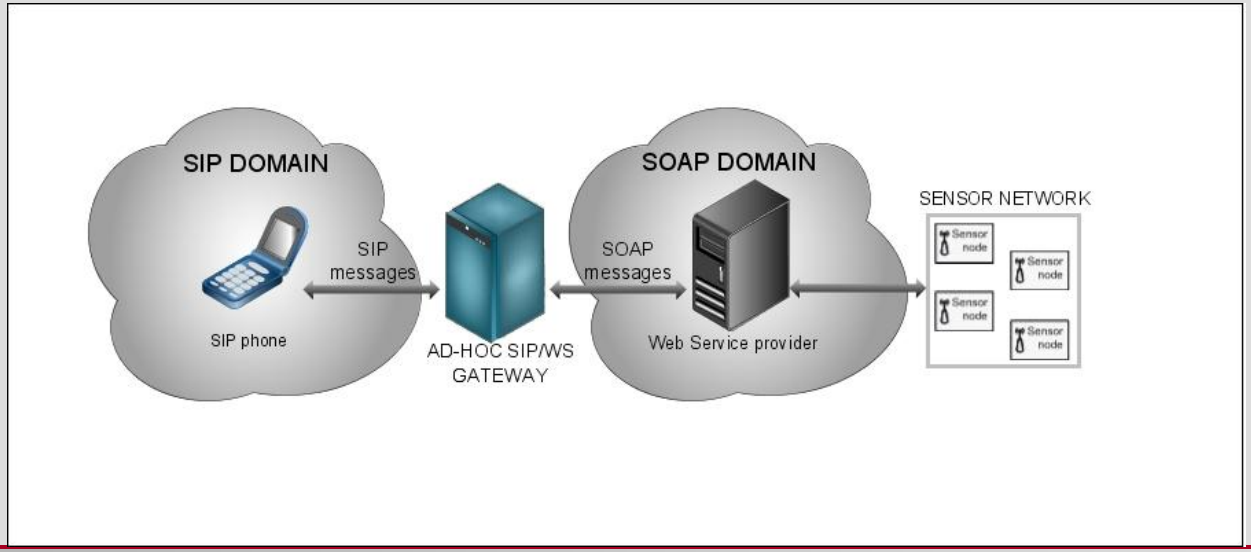
Primjer

- **Pristup mreži osjetila protokolom uspostave sjednice**
  - Mreža osjetila prati temperaturu u laboratoriju
  - Na mrežu osjetila spojeno je računalo koje izlaže SOAP uslugu kroz koju su dostupne očitane vrijednosti
  - Korisnik želi pristupiti usluzi mreže osjetila koristeći SIP telefon (ne podržava SOAP protokol)



Primjer

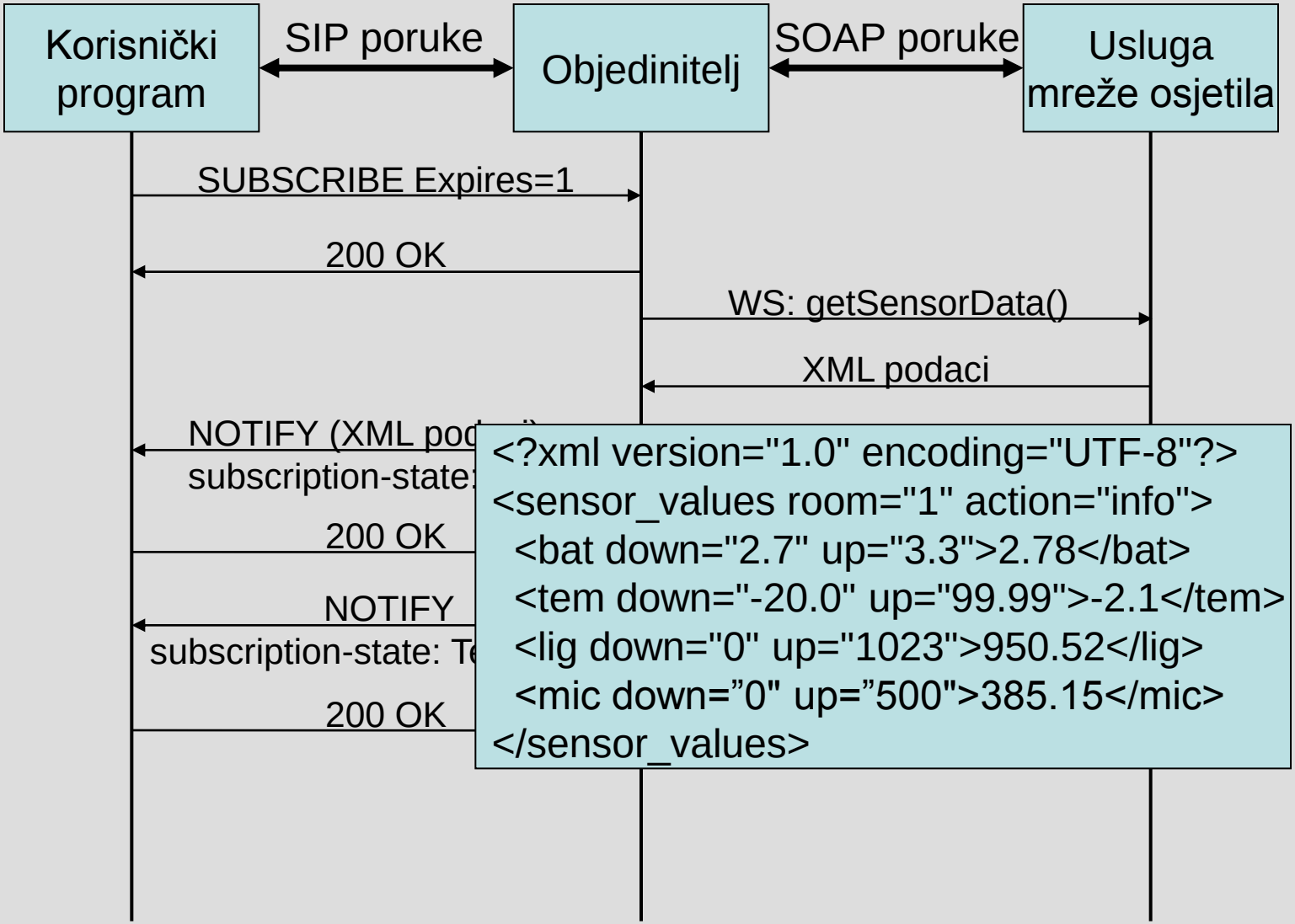
- **Moguće rješenje**
  - Izgraditi sustav koji prima zahtjeve SIP telefona i u skladu s preddefiniranim pravilima oblikuje SOAP poruke i dohvaća podatke od usluge mreže osjetila
  - Dohvaćene podatke sustav treba dojaviti SIP telefonu koristeći protokol uspostave sjednice



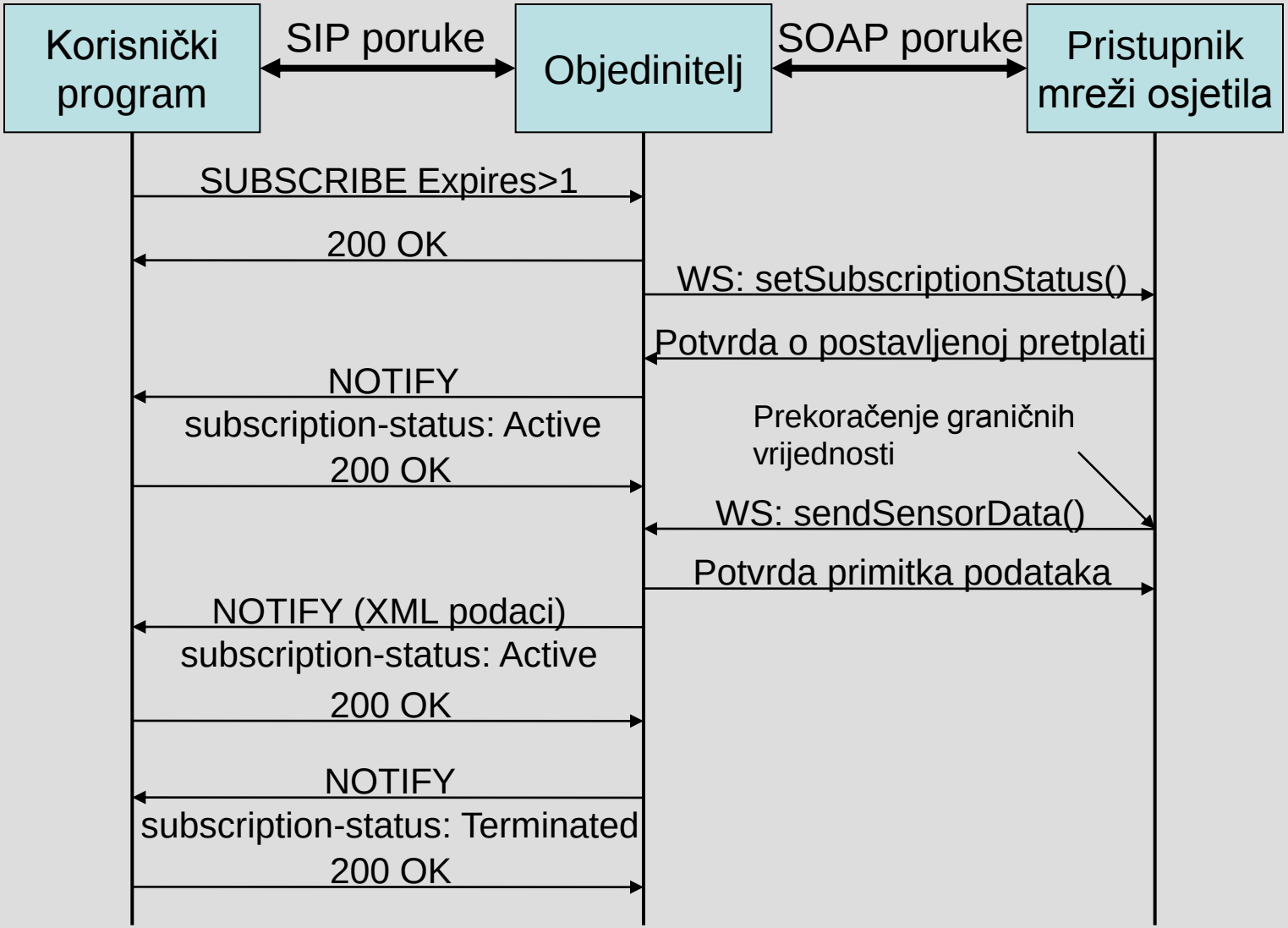
Primjer

- **Usluga mreže osjetila podržava dva načina rada**
  - Jednokratan dohvat mjernih vrijednosti
  - Pretplata na događaj

Jednokratan dohvat vrijednosti



Pretplata na događaj



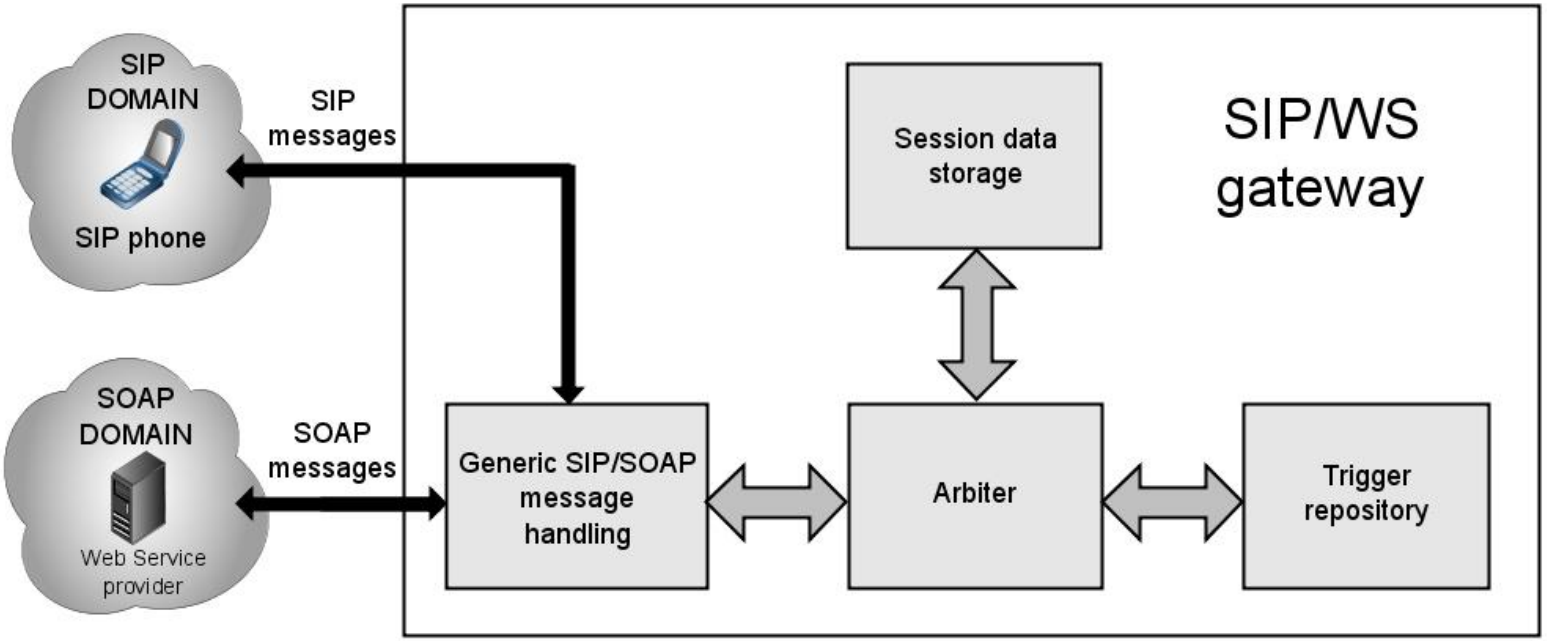
SIP/WS posrednik

- **Izgrađeni sustav (objedinitelj) moguće je iskoristiti samo za pristup usluzi mreže osjetila**
- **Ograničenja sustava**
  - Jednosmjerna komunikacija (SIP klijent, SOAP poslužitelj)
  - Protokol između klijenta, poslužitelja i objedinitelja je izravno određen implementacijom objedinitelja (eventualne promjene zahtijevaju gašenje sustava, promjene u izvornom kodu i ponovno prevođenje)
    - Unaprijed poznat format SIP i SOAP poruka

SIP/WS posrednik

- **Zahtjevi za SIP/WS posrednik opće namjene**
  - Obosmjerna komunikacija
  - Proizvoljan broj sudionika u određenoj *aplikaciji*
  - Proizvoljan broj istovremeno aktivnih aplikacija
  - Dodavanje i uklanjanje aplikacija bez gašenja sustava (*runtime*)
  - Definiciju protokola pojedine aplikacije zadaje korisnik sustava (*trigger*)

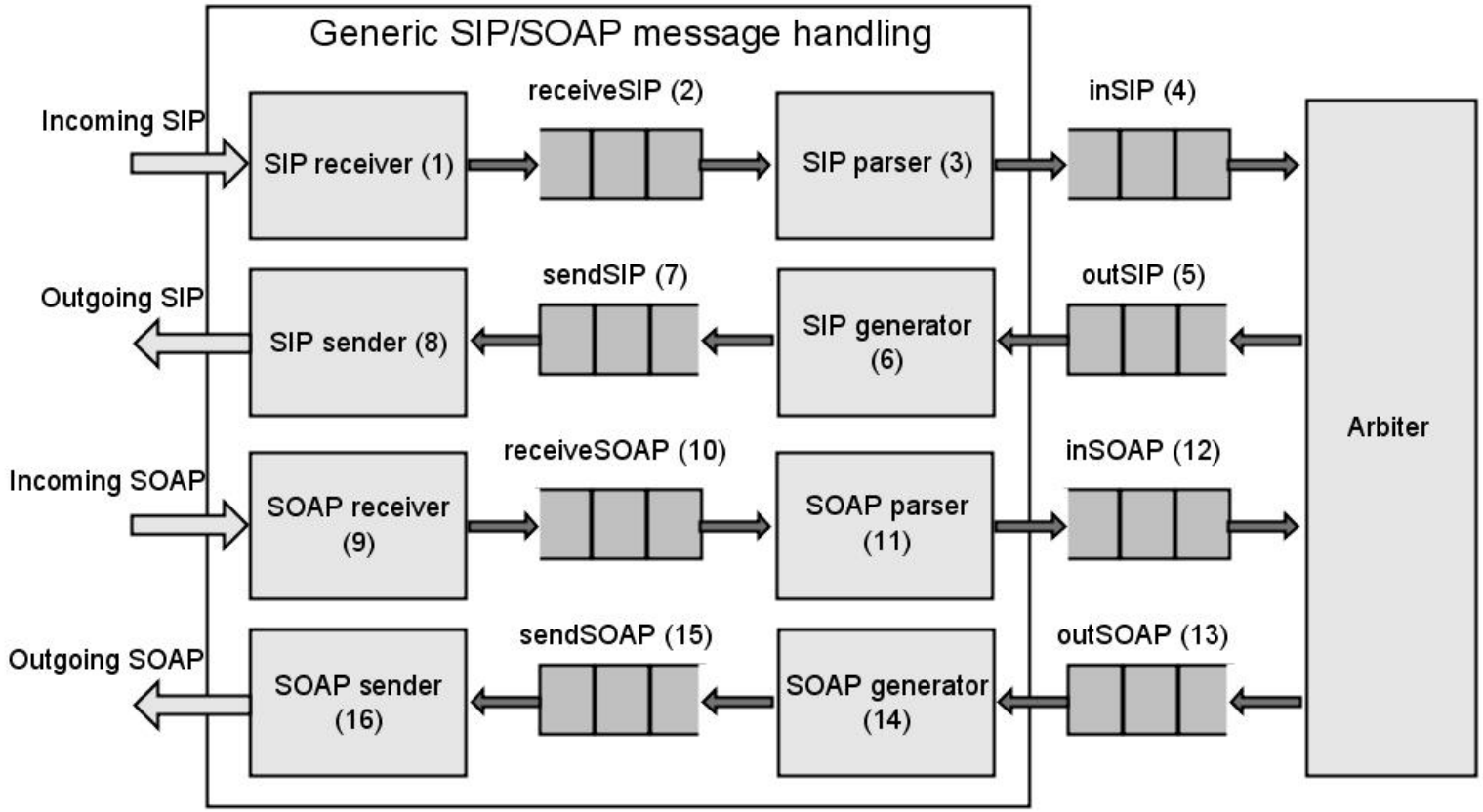
Arhitektura SIP/WS posrednika



Modul za rukovanje porukama

- **Zadužen za parsiranje i generiranje SIP/SOAP poruka**
- **Upravlja mrežnim sredstvima računala domaćina**
- **Razmjenjuje opisnike poruka sa upravljačkom jedinicom (*arbiter*) kroz redove**

Modul za rukovanje porukama



Modul za pohranu podataka

- **Aplikacijski podaci**
  - Uvijek sadrži identifikator aplikacije (globalno jedinstven)
  - Primjer: statistika korištenja određene usluge
- **Podaci o sjednici**
  - Svaki novi klijent u određenoj aplikaciji otvara novu sjednicu
  - Isti klijent može nakon zaključenja sjednice u nekom vremenskom trenutku započeti novu sjednicu
  - Uvijek sadrži identifikator sjednice (globalno jedinstven)
  - Primjer: trajanje pretplate na događaj

Spremnik opisnika aplikacija

- **Opisnik aplikacije (trigger) zadaje se posebno za svaku aplikaciju u sustavu**
- **Definira što posrednik treba napraviti nakon što primi određenu poruku**
  - Slanje odgovora i zahtjeva drugim uslugama koristeći modul za rukovanje porukama
  - Promjena stanja sjednice ili aplikacije (promjena struktura podataka u modulu za pohranu podataka)
- **Novi opisnik aplikacije moguće je dodati u spremnik opisnika aplikacija tijekom rada sustava**

Spremnik opisnika aplikacija

- **Opisnici aplikacija imaju uniformno sučelje**
  - *getSessionId*
    - Iz poruke dohvaća identifikator sjednice
  - *match*
    - Za danu poruku vraća listu imena *akcija* koje sustav treba izvršiti
    - Akcije su definirane u opisniku aplikacije
    - Na razini sustava su unaprijed definirane akcije *NOP* i *PURGE* (briše podatke vezane uz sjednicu iz spremnika podataka)
  - *activate*
    - Za zadanu listu akcija vraća listu opisnika izlaznih poruka koje sustav treba poslati i listu operacija nad spremnikom podataka

Upravljačka jedinica (*arbiter*)

- **Za svaku primljenu poruku pretražuje spremnik opisnika aplikacija za odgovarajući opisnik (neprazna *match* lista)**
- **Poziva odgovarajuću *activate* metodu**
  - Opisnike izlaznih poruka predaje modulu za rukovanje porukama
  - Izvodi zadane operacije nad spremnikom podataka

Primjena SIP/WS posrednika

- **Sustav se može postaviti na jedno računalo ili u raspodijeljenu okolinu**
  - Moduli su međusobno slabo povezani
  - Za veću propusnost, moguće je paralelno izvoditi veći broj SIP/SOAP parsera/generatora i upravljačkih jedinica uz sinkronizaciju pri pristupu dijeljenim redovima
- **Definicija opisnika aplikacije**
  - Ručna implementacija uniformnog sučelja opisnika
  - Opis aplikacije specijaliziranim jezikom (DSL – Domain Specific Language) i automatizirano generiranje opisnika

Primjena SIP/WS posrednika

- **PCCL (Protocol Conversion and Coordination Language)**
  - Ericsson Nikola Tesla
  - Zasnovan na XML-u
  - Iz opisa aplikacije PCCL-om generira se izvodivi opisnik aplikacije u skladu s uniformnim sučeljem SIP/WS posrednika
  - Generirani opisnik aplikacije može se postaviti na SIP/WS posrednik

Pristup usluzi mreže osjetila protokolom uspostave sjednice kroz SIP/WS posrednik

- **Definicija usluge mreže osjetila**

```
<WS name="ws1">
    <wsdl>
      http://192.168.0.2/WS/Service.asmx?WSDL
    </wsdl>
    <location>
      http://192.168.0.2/WS/Service.asmx
    </location>
    <host>
      192.168.0.2
    </host>
    <ID soapAction="setSubscriptionStatus" />
    <ID soapAction="getSensorData" />
</WS>
```

- **U ostatku definicije niz *ws1* predstavlja identifikator usluge mreže osjetila**

Pristup usluzi mreže osjetila protokolom uspostave sjednice kroz SIP/WS posrednik

- ***Action* elementi**
  - Definiraju koje izlazne poruke i s kojim argumentima sustav treba generirati
  - Definiraju promjene nad spremnikom podataka

- **Na osnovi *action* elemenata generira se *activate* metoda opisnika aplikacije**

Pristup usluzi mreže osjetila protokolom uspostave sjednice kroz SIP/WS posrednik

```
<action name="OneTimePull" inType="SIP">
  <get type="string" location="header:method">
       "?room="$mySession.roomID$" "
  </get>

  <set state="OTP" />

  <send protocol="SIP" type="OK"/>

  <send protocol="SOAP" name="getSensorData" type="Request"
            service="ws1">
    <arg name="roomID" type="string">
         $mySession.roomID$
    </arg>
  </send>
</action>
```

Pristup usluzi mreže osjetila protokolom uspostave sjednice kroz SIP/WS posrednik

- ***Protocol* elementi**
  - Definiraju koje akcije treba izvesti za primljenu poruku
  - Odabir akcija zasniva se na sadržaju poruke i na trenutnom stanju posrednika (sadržaj spremnika podataka)
- **Na osnovi *protocol* elemenata generira se *match* metoda opisnika aplikacije**

Pristup usluzi mreže osjetila protokolom uspostave sjednice kroz SIP/WS posrednik

```
<protocol name="SIP">
   <message type="SUBSCRIBE">
      <condition>
        <if>
          <EQ left="header:expires" right="1" type="int" />
          <exec>
            OneTimePull
          </exec>
        </if>
        <elif>
          <GT left="header:expires" right="1" type="int" />
          <exec>
            ...
          </exec>
        </elif>
     </condition>
  </message>
</protocol>
```

Zaključak

- **Specijalizirano rješenje za pristup mreži osjetila protokolom uspostave sjednice**
  - Monolitan sustav
  - Zbog uskog područja primjene, moguće brojne optimizacije
  - Razvoj sustava trajao oko mjesec dana
- **SIP/WS posrednik opće namjene**
  - Zbog općenitosti, unosi veća kašnjena od specijaliziranog sustava
  - PCCL definicija aplikacije pristupa mreži osjetila ima 314 linija XML-a (vrijeme “razvoja” ~ 15 minuta)

Literatura

- A. Ray, R. Cleveland, **“Architectural interaction diagrams”**, Proceedings of ICSE, 2003.
- A. Ray, R. Cleveland, **“Formal Modeling of Middleware-based Distributed Systems”**, Electronic Notes in Theoretical Computer Science, 2004.
- I. Budiselić, G. Delač, D. Šego, T. Štefanec, **"SIP/WS interworking triggering gateway“,** technical report for SC2007, ETK