DB-all.e

- It is based on physical principles, that is, the data it contains are defined in terms of **homogeneous** and **consistent** physical data. For example, it is impossible for two incompatible values to exist in the same point in space and time.
- It can manage **fixed stations** and **moving stations** such as **airplanes** or **ships**.
- It can manage both **observational** and **forecast** data.
- It can manage data along **all three dimensions in space**, such as data from soundings and airplanes.
- Report information is preserved. It can work based on physical parameters or on report types.

The Arkimet archiving system

- Set di tool per archiviazione e distribuzione di dati ambientali
- Accesso locale (filesystem), remoto (HTTP) omogeneo tramite CLI
- Integrità dei dati: sono trattati come una stringa binaria opaca, in sola lettura e mai modificata
- E' possibile estendere facilmente i formati supportati (attualmente GRIB, BUFR, ODIMH5)
- Deploy molto semplice e veloce
- Arkiweb: interfaccia web http://www.smr.arpa.emr.it/arkiweb
- Sviluppato dal SIMC Licenza GPLv2+
- http://svn.smr.arpa.emr.it/arkimet/arkimet/trunk

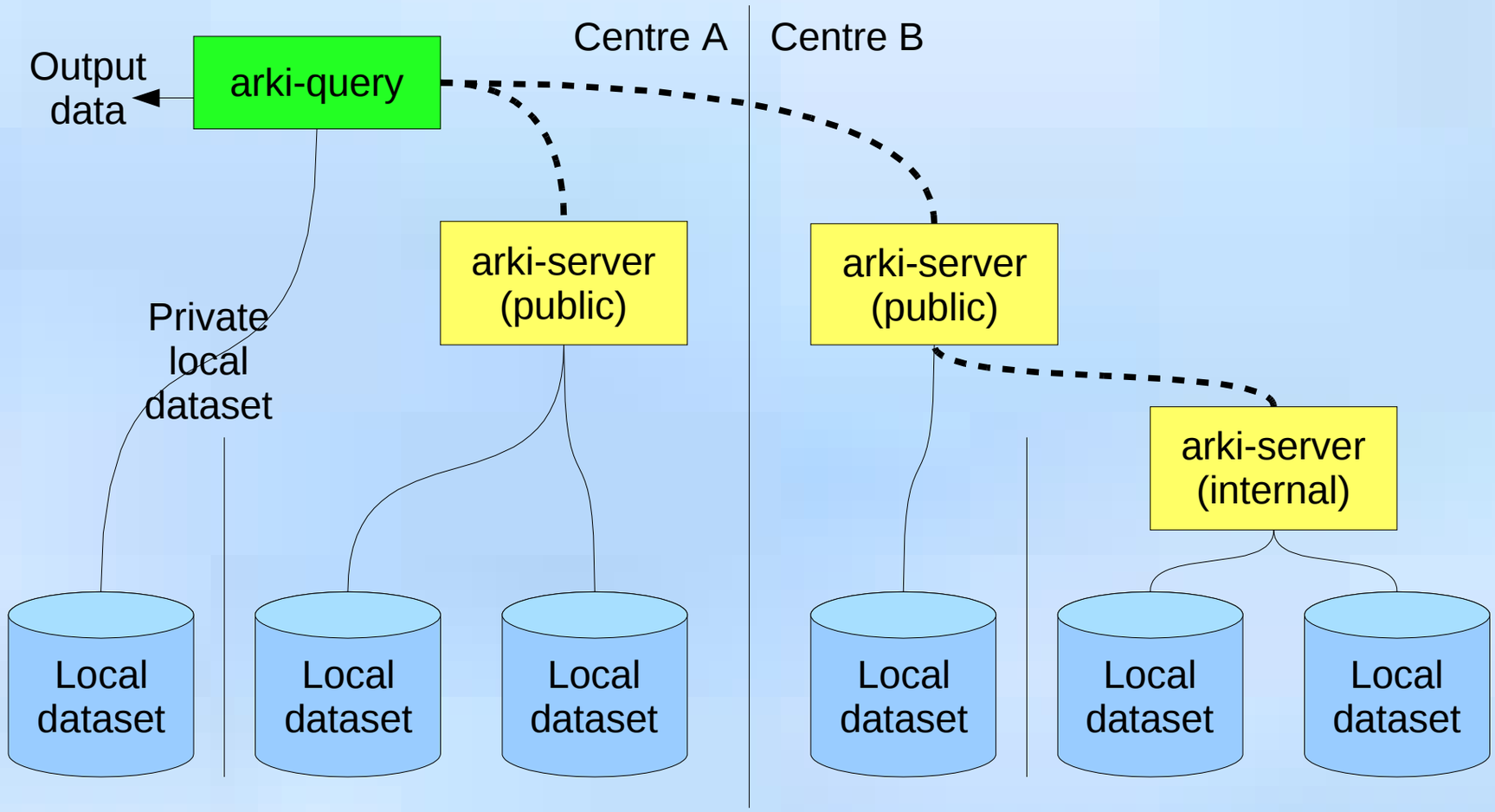
Arkimet: come lavora

- Data is examined and metadata are extracted
- Data and metadata are acquired into datasets
- Datasets are self-contained collections of homogeneous information
- Datasets store data, metadata, and also summaries of the data
- A summary can be used to explore the contents of a dataset, or the output of a query, without extracting the data

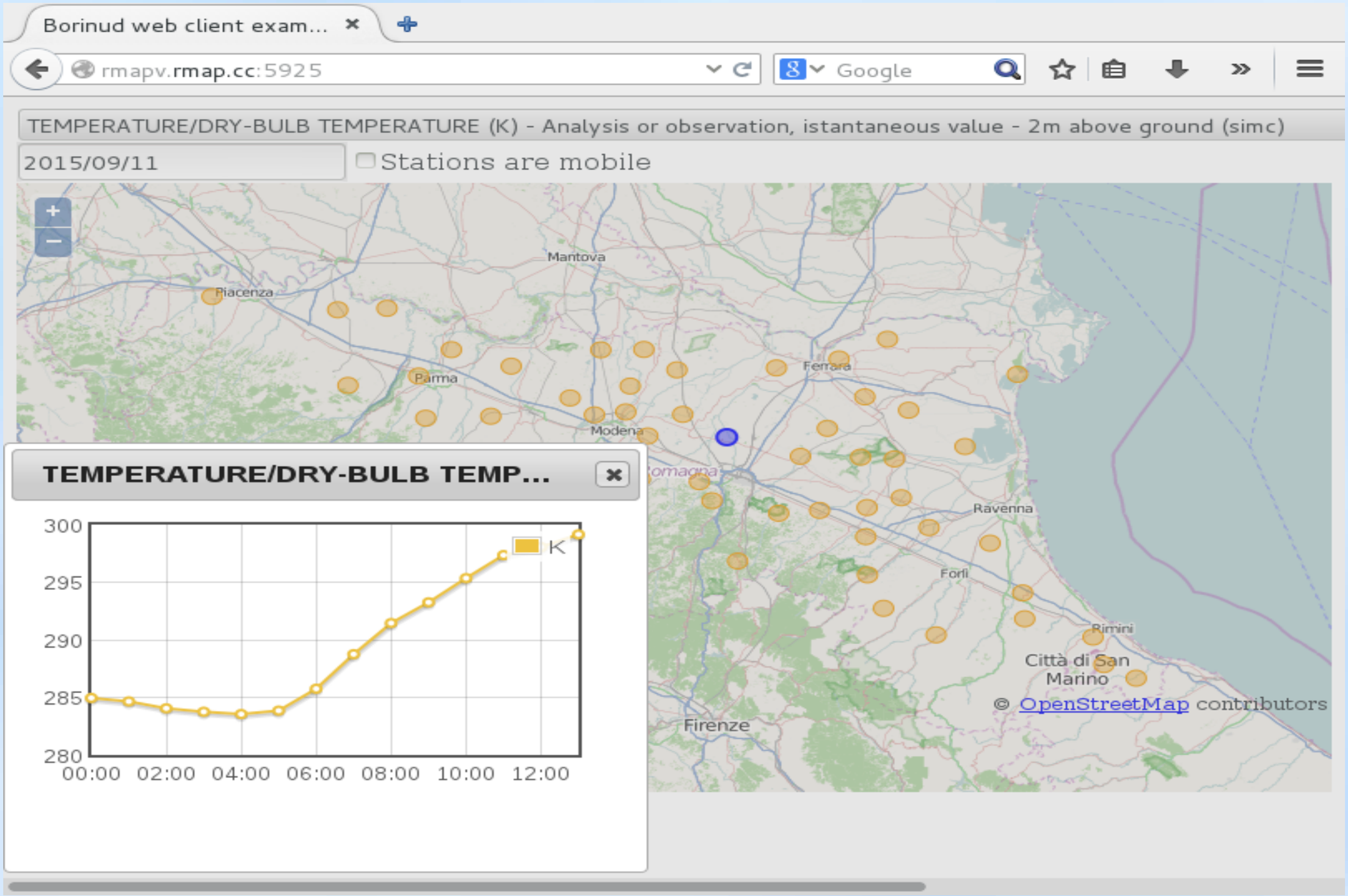
Arkimet: distributed

- Data is accessible locally and remotely, in the same way
- Remote access uses a client-server model, over standard HTTP
- Any centre / unit can deploy their own
- Server can share local and remote datasets
  - It is possible to create a public front-end server that aggregates several internal servers

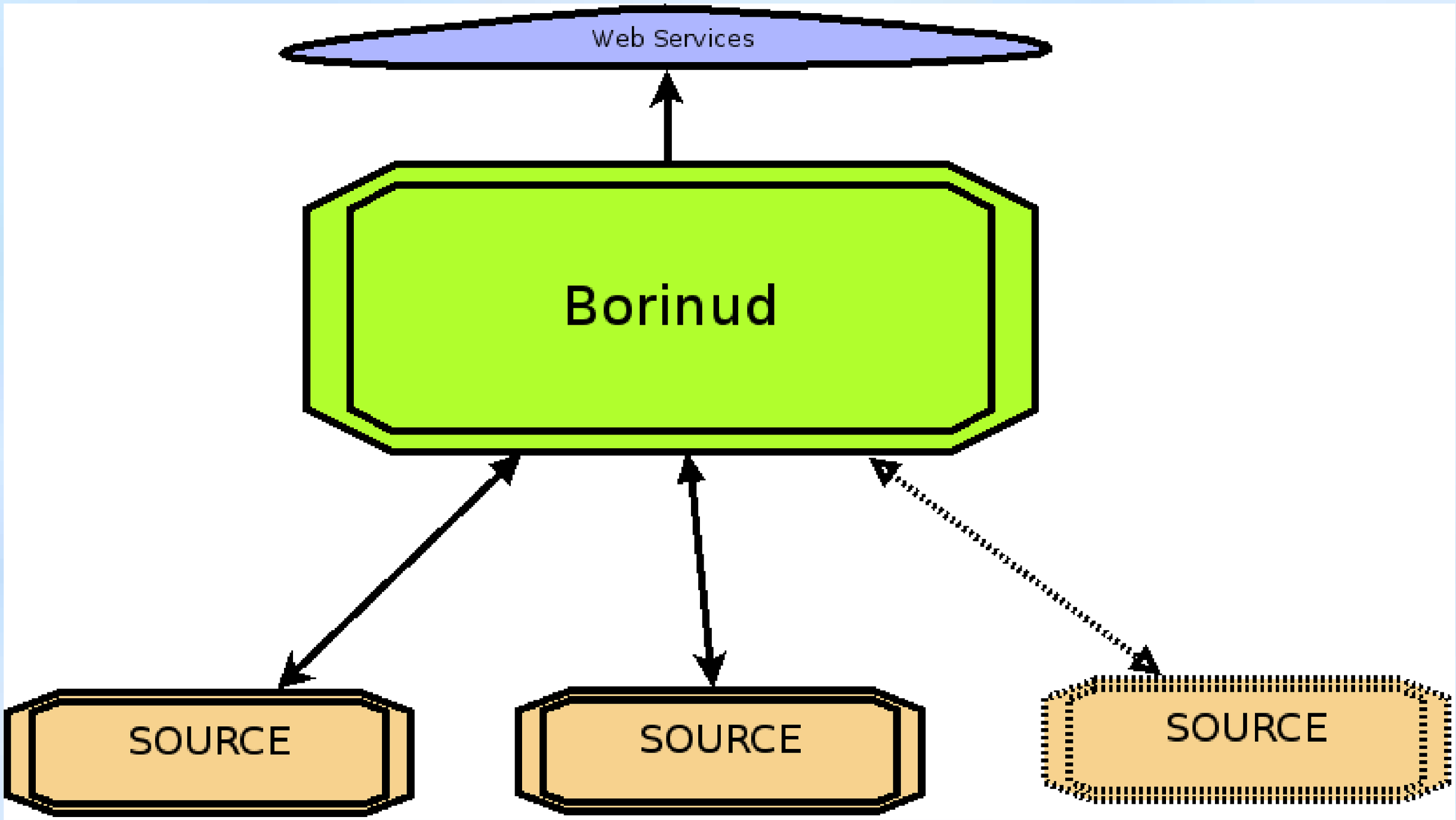
Arkimet: distributed



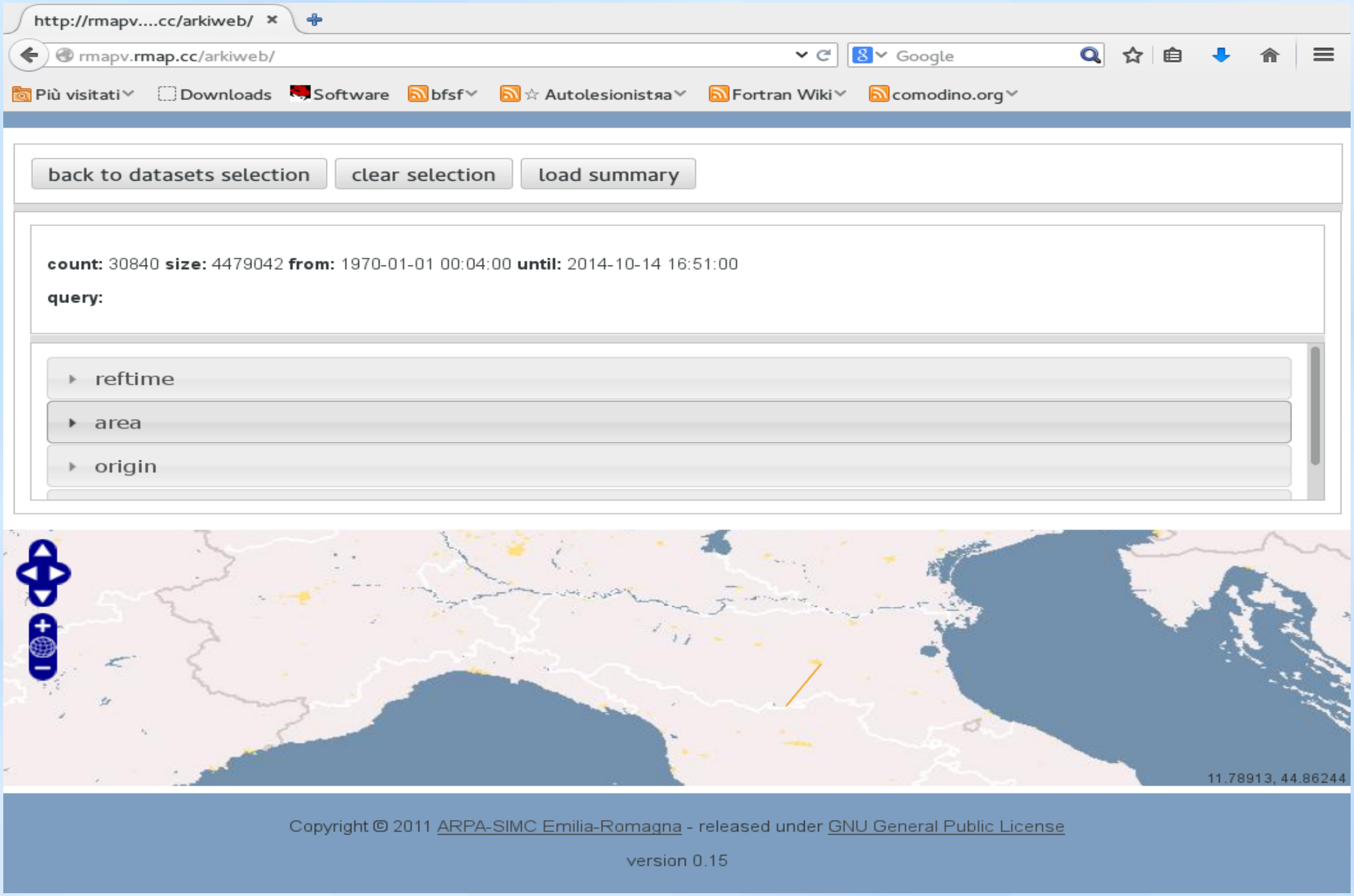
Borinud: DB-All.e web and web service



Borinud merge dati



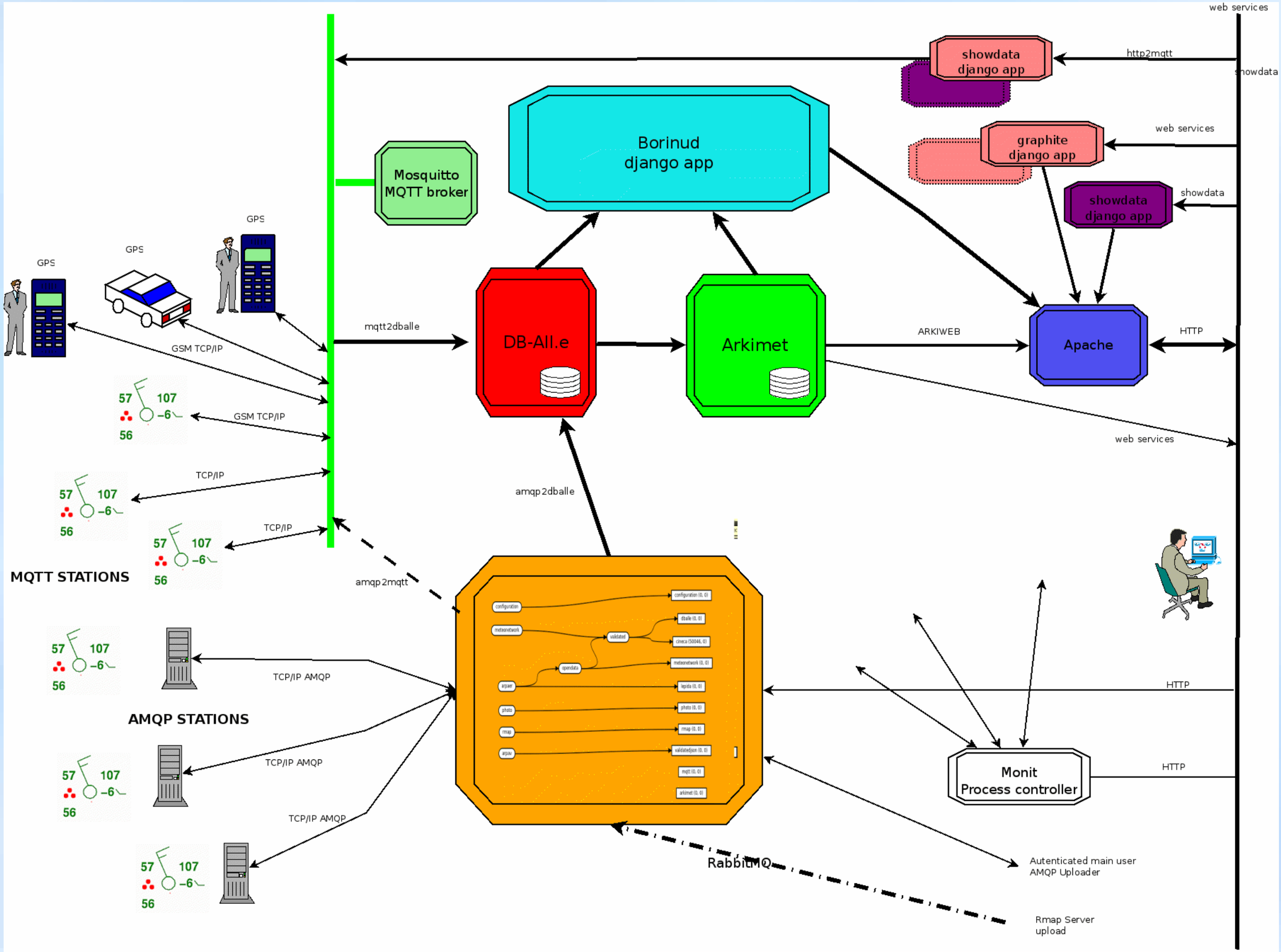
Arkiweb


http://rmapv....cc/arkiweb/
rmapv.rmap.cc/arkiweb/
Google
Più visitati
Downloads
Software
bfsf
Autolesionistяa
Fortran Wiki
comodino.org
back to datasets selection
clear selection
load summary
count: 30840 size: 4479042 from: 1970-01-01 00:04:00 until: 2014-10-14 16:51:00
query:
reftime
area
origin
11.78913, 44.86244
Copyright © 2011 ARPA-SIMC Emilia-Romagna - released under GNU General Public License
version 0.15

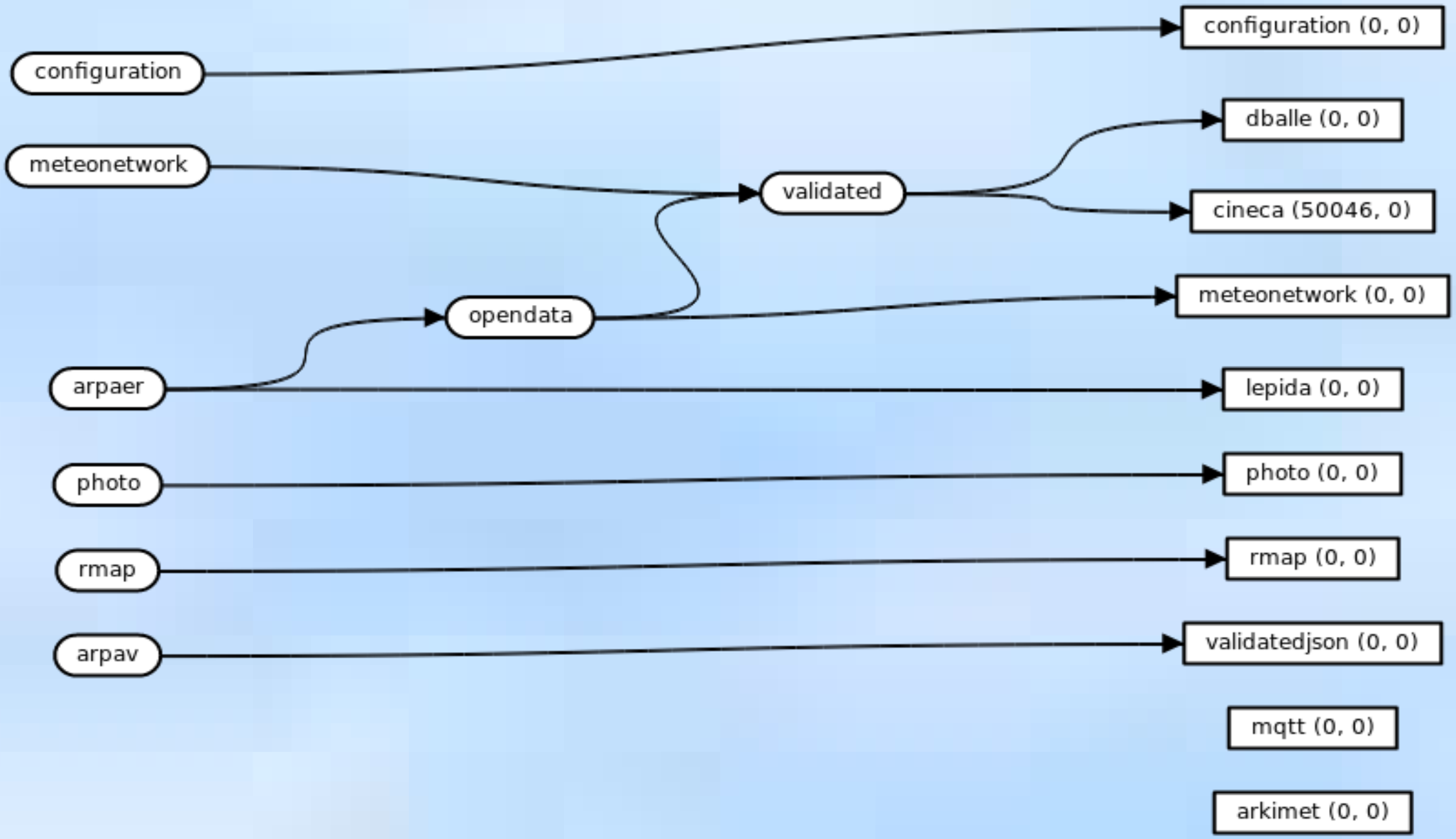
Libsim

- Libsim comprende quattro gruppi di moduli di utilità in Fortran 90:
  - libsim_base definisce moduli e classi di uso generale in applicazioni scientifiche, come la gestione di errori in esecuzione, la gestione di dati georeferenziati, di coordinate temporali, ecc.
  - libsim_grib definisce una serie di classi ad alto livello stratificate sopra la libreria ECMWF grib_api per gestire l'I/O di file in formato grib.
  - libsim_vol7d definisce una serie di classi per facilitare l'elaborazione di dati osservativi idro-meteo, includendo metodi per la loro importazione da database tipo DbAll-e
  - libsim_volgrid6d definisce una serie di classi per facilitare l'elaborazione di dati idro-meteo su grigliati georeferenziati, compresa la trasformazione in griglie di tipo diverso e in oggetti di tipo vol7d.

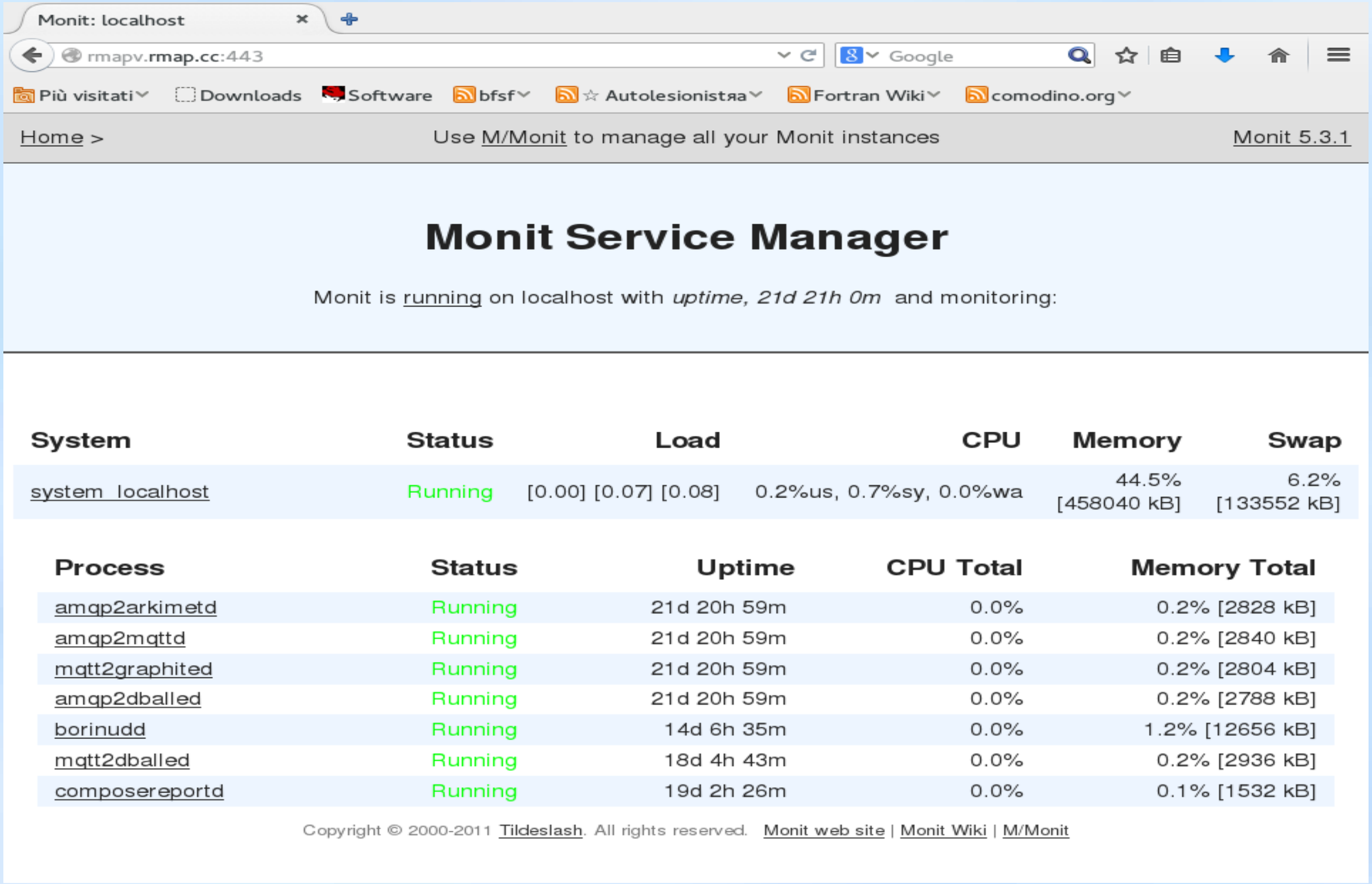
Flusso dati e processi



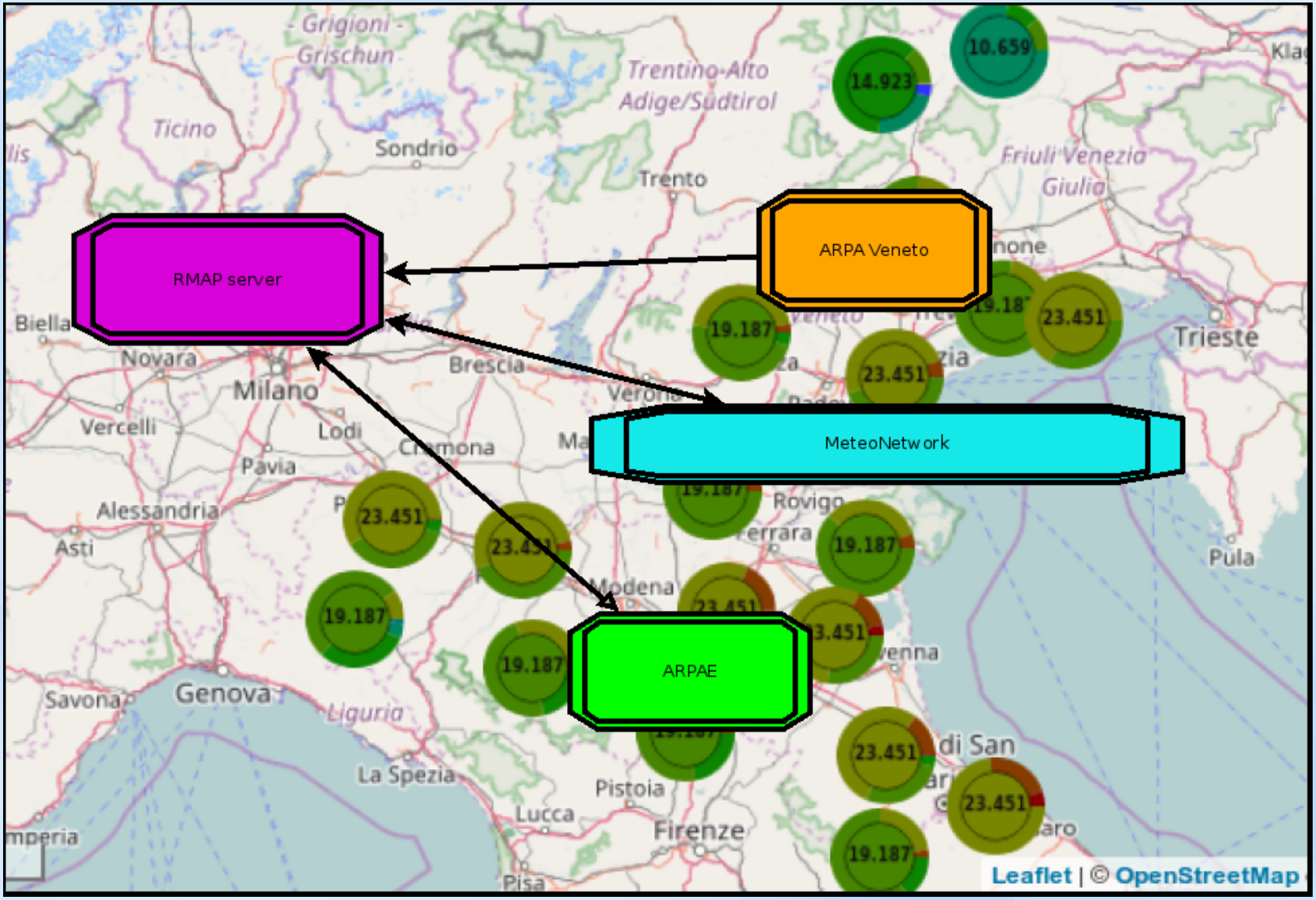
Exchange e queue di RabbitMQ



Monit


Monit: localhost
rmapv.rmap.cc:443
Google
Più visitati
Downloads
Software
bfsf
Autolesionistяa
Fortran Wiki
comodino.org
Home >
Use M/Monit to manage all your Monit instances
Monit 5.3.1
Monit Service Manager
Monit is running on localhost with uptime, 21d 21h 0m and monitoring:
System
Status
Load
CPU
Memory
Swap
system_localhost
Running
[0.00] [0.07] [0.08]
0.2%us, 0.7%sy, 0.0%wa
44.5% [458040 kB]
6.2% [133552 kB]
Process
Status
Uptime
CPU Total
Memory Total
amqp2arkimetd
Running
21d 20h 59m
0.0%
0.2% [2828 kB]
amqp2mqttd
Running
21d 20h 59m
0.0%
0.2% [2840 kB]
mqtt2graphited
Running
21d 20h 59m
0.0%
0.2% [2804 kB]
amqp2dballed
Running
21d 20h 59m
0.0%
0.2% [2788 kB]
borinudd
Running
14d 6h 35m
0.0%
1.2% [12656 kB]
mqtt2dballed
Running
18d 4h 43m
0.0%
0.2% [2936 kB]
composereportd
Running
19d 2h 26m
0.0%
0.1% [1532 kB]
Copyright © 2000-2011 Tildeslash. All rights reserved. Monit web site | Monit Wiki | M/Monit

Raccolta Dati attualmente operativa



Total Physical Source Lines of Code

non-blank, non-comment lines

cost estimates include design, coding, testing (including integration and testing), documentation (both for users and for programmers), and a wrap rate for corporate overhead (to cover facilities, equipment, accounting, and so on)

- Python (stazione + server+GUI)
  - Total Physical Source Lines of Code (SLOC) = **15,626**
  - Total Estimated Cost to Develop = **$ 484,375**
- Firmware Stima
  - Total Physical Source Lines of Code (SLOC) = **131,352**
    - Solo il codice sviluppato per Stima
      - Total Physical Source Lines of Code (SLOC) = **12,763**
      - Total Estimated Cost to Develop = **$ 391,644**
- Mqtt2bufr
  - Total Physical Source Lines of Code (SLOC) = **730**
  - Total Estimated Cost to Develop = $ **19,415**

Il firmware STIMA

Stazioni fisse o mobili

È possibile installare sia stazioni fisse, la cui posizione non cambia nel tempo, sia stazioni mobili, sia terrestri che marine. Per aggiornare la posizioni delle stazioni mobili viene utilizzato un GPS che può essere

- a bordo del modulo Stima
- a bordo di un dispositivo android.

Il firmware STIMA

Differenti tipologie di rete

La configurazione della rete può essere differente a seconda delle esigenze:

- configurazione a stella (moduli master e base) con un broker al centro
- configurazione ad albero sia via cavo (modulo master + base) che via radio

con la possibilità di utilizzare moduli radio di maggiore potenza (~1Km in aria libera) è possibile prevedere coperture di un territorio di ampia superficie.

Il firmware STIMA

Salvataggio locale dei dati

I dati possono essere pubblicati in real time e/o salvati localmente.

- salvataggio dei dati su SD formattata FAT;
- i file vengono frammentati a una dimensione prefissata per farne circa uno al giorno e numerati da 000 a 999; i dati salvati hanno un flag che indica se i dati sono stati già pubblicati correttamente su MQTT;
- i file che devono essere controllati per possibili reinvii hanno postfisso .que e quelli che hanno tutti i dati già inviati hanno postfisso .don.

Il firmware STIMA

Salvataggio locale dei dati; funzionalità:

- salvataggio dati su SD almeno per due anni con campionamenti ogni 5s (un parametro)
- reinvio automatico al server dei dati salvati ma non pubblicati correttamente sul server
- ottimizzazione dei tempi in quanto solo i file che contengono dati da inviare vengono letti per selezionare i dati da reinviare
- i dati possono essere riletti su un normale PC estraendo la SD

Il firmware STIMA

Messaggistica di diagnostica

C'è la possibilità di ottenere una ampia messaggistica di diagnostica per la soluzione dei problemi

attiva di default può essere disabilitata

Il firmware STIMA

Configurazione

Le versioni delle configurazioni vengono verificate e quando il firmware non è retrocompatibile il modulo resta in attesa di una nuova configurazione

Si può forzare la configurazione tramite un apposito ponticello sulla board Stima-I2C

Le configurazioni vengono subito verificate: non è possibile configurare un modulo con dei sensori non corretti o non funzionanti

Il firmware STIMA

Operazioni di mantenimento periodiche

Il software effettua periodicamente tutte le funzioni di mantenimento necessarie a un corretto funzionamento:

- DHCP
- sincronizzazione dell'orologio interno con una sorgente esterna
- la gestione dei pacchetti per il mantenimento dei protocolli su TCP/IP o via radio

Tutti i firmware hanno attivo un watchdog hardware che evita blocchi permanenti dovuti a malfunzionamenti su eventi improbabili; ogni 8 secondi quindi il watchdog deve essere reinizializzato per evitare un reset del microcontrollore.

Il firmware STIMA

Orologio di riferimento

- Una base dei tempi precisa è richiesta nel caso in cui sia necessario salvare i dati localmente (su SD) nel caso la connessione utilizzata per pubblicare i dati sul server (broker) non sia considerata stabile.
- Se invece la connessione (trasporto) viene considerata stabile (o non sia necessario recuperare i dati in caso di fault) un preciso orologio di riferimento non è necessario e il tempo di riferimento verrà aggiunto automaticamente dal server in tempo reale alla pubblicazione del dato.

Ci sono diversi sistemi per avere un orologio di riferimento preciso sui moduli Stima.

Il firmware STIMA

Crittografia

Qualora il trasporto non sia considerato sicuro (via radio) viene utilizzata la crittografia per garantire riservatezza e autenticità.

Per ora il sistema è molto semplice ed utilizza AES con chiavi statiche.

Il firmware STIMA

Attenzione ai consumi energetici

Attenzione è stata posta alla limitazione dei consumi

- Quando possibile i microcontrollori e i sensori vengono messi in sleep e sono alcuni interrupt a risvegliare il sistema

Questo agevola l'utilizzo con batterie dei sistemi a basso consumo quali il modulo Stima-satellite che funziona con un modulo radio.

Il firmware STIMA

Integrazione con la domotica

Per quello che è stato possibile si è cercato di integrarsi con gli standard della domotica (MQTT)

Tutti i moduli possono essere utilizzati anche da attuatori on/off (fino a 4 relay)

è molto semplice aggiungere altre funzionalità tramite remote procedure in formato json su tutti i trasporti

Le librerie utilizzate dal firmware

aJson

aJson è un tentativo di portare una completa implementazione di JSON a Arduino.

E' basata su cJSON, ridotta di dimensione.

La libreria è stata adattata per ridurre l'uso della memoria e implementare alcuni tipi dato non supportati.

Le librerie utilizzate dal firmware

JsonRPC

implementa un sottoinsieme del protocollo JSON-RPC.

PubSubClient

Fornisce un client per semplici publish/subscribe scambiando messaggi con un server che supporta MQTT (broker).

Modificata per funzionare oltre che con ethernet anche con GSM/GPRS.

Le librerie utilizzate dal firmware

Arduino_uip

Implementa le stesse API della Arduino Ethernet library ma utilizzando ENC28J60 come chip per la comunicazione ethernet.

Supporto completo per le connessioni TCP e UDP persistenti (streaming) (Client e Server), ARP, ICMP, DHCP and DNS.

Sviluppata da Norbert Truchsess derivando dallo stack uIP di Adam Dunkels. Oltre a creare una implementazione completamente open permette di utilizzare ENC28J60, chip molto più economico di quelli con stack IP incluso.

Le librerie utilizzate dal firmware

RF24 / RF24Network

ultima versione elaborata da TMRh20 che ci è risultata essere la più stabile e con più opzioni oltre a funzionare con Arduino e Rpi.

- OSI Network Layer ottimizzato per radio nRF24L01(+) 2.4GHz ISM. Con la tipologia di modulo da noi utlizzato si possono coprire distanze di circa 50m in aria libera.
- Host Addressing: ogni nodo ha un indirizzo logico nella rete locale.
- Message Forwarding: i messagi possono essere mandati da un nodo a qualsiasi altro nodo senza limite al numero di "salti" che il messaggio deve fare.
- Ad-hoc Joining: un nodo può entrare a far parte della rete senza nessun cambiamento alla configrazione dei nodi già esistenti.

Le librerie utilizzate dal firmware

sim800

Questa libreria è stata sviluppata ex novo in quanto le funzionalità richieste non sono disponibili in nessun altra libreria di gestione dei moduli sim800/sim900. La libreria implementa:

- TCP/IP transparent mode con le API Etherlib
- http in modalità nativa sim800
- utilizzo dell'RTC interno alla sim800

Le librerie utilizzate dal firmware

SensorDriver

Libreria di "driver" per la gestione dei sensori. Di questa libreria esistono attualmente due versioni, una in C++ e una in python.

Porta la gestione della sensoristica ad un livello di astrazione più alto.

Aggiungere un nuovo tipo di sensore consiste nell'estendere una classe con quattro metodi per effettuare la lettura di quello specifico sensore

int setup (int address);

effettua eventuali settaggi necessari al funzionamento del sensore

int prepare (unsigned long* waittime);

impartisce al sensore il comando per effettuare una singola misurazione torna il tempo in millisecondi di attesa necessario

int get (int* value);

torna i valori della misurazione

aJsonObject* getJson();

torna i valori misurati in formato json

Le librerie utilizzate dal firmware

Time

Time fornisce un orologio software; l'orologio può essere sincronizzato con sorgenti esterne per mantenere l'orologio preciso. Nel nostro caso utilizziamo

- NTP se disponibile ethernet
- RTC della sim800
- RTC del DS1307 tramite I2C.

TimeAlarms

TimeAlarms unitamente a Time esegue funzioni a istanti di tempo specificati, unatantum o periodicamente.

Le librerie utilizzate dal firmware

SdFat

SdFat è una libreria per Arduino che supporta FAT16 and FAT32 file systems su SD cards standard o ad alta capacità (SD/SDHC flash cards)

SdFat supporta la creazione di file, cancellazione, read, write, oltre al troncamento. SdFat supporta l'accesso a subdirectories, creazione, cancellazione si subdirectories. Supporta Long File Names e usa la libreria Arduino SPI

Le librerie utilizzate dal firmware

AESLib

Questa libreria implementa la crittografia tramite AES per arduino.

La crittografia nel nostro firmware è utilizzata per le comunicazioni con nrf24, ma il livello di sicurezza non è ancora elevato e l'implementazione è incompleta.

Le librerie utilizzate dal firmware

YwrobotLiquidCrystal_I2C

Libreria per la gestione del display LCD tramite I2C

Supporta gran parte delle funzioni listate nella specifica ufficiale per i display LCD

Configurazione a tempo di compilazione

- Il firmware Stima fa un uso intensivo delle direttive del **preprocessore C**

- Definendo delle variabili è possibile ottenere da un unico codice diversi firmware per i vari moduli

- Il file **rmap_config.h** situato in sketchbook/rmap/rmap comanda quasi tutte le opzioni e le configurazioni hardware. Sono presenti i **template** da utilizzare per ottenere i firmware per i moduli qui presentati.

Le funzioni principali del firmware

Nel main loop:

- **mgrserialjsonrpc**: gestione della porta seriale per jsonrpc
- **mgrrf24jsonrpc**: gestione della comunicazione radio e jsonrpc
- **mgrmqtt**: gestione del protocollo MQTT e jsonrpc
- **mgrethserver**: gestione del server tcp/ip per jsonrpc

La funzione **mgrjsonrpc** è la funzione comune per gestire il protocollo json-RPC.

Se il modulo è “attivo” periodicamente viene eseguita la funzione **Repeats** che svolge le operazioni di interrogazione dei sensori e pubblicazione dei dati.

STIMA software

- http://rmap.cc
- http://liste.raspibo.org/wws/subscribe/meteo
- https://github.com/r-map

SOS - Introduzione

- Standard OGC dal 2007
- http://www.opengeospatial.org/standards/sos
- Standard che definisce l'interfaccia di un servizio web per l'interrogazione di osservazioni, metadati dei sensori e rappresentazione delle caratteristiche osservate
- Tre servizi di base:
  - **GetCapabilities**: informazioni sul servizio e sui sensori disponibli
  - **DescribeSensor**: metadati del sensore (*SensorML*)
  - **GetObservation**: valori misurati dai sensori (*Observations and Measurements*)
- Servizi opzionali

SOS – parole chiave

- **Procedure** è ciò che produce l'osservazione
  - Sensore
  - Postprocessamento (e.g. media, massima, minima)
- **Observed property** è la proprietà osservata
- **Feature Of Interest** è l'oggetto georeferenziato che viene misurato
  - Nel nostri casi, generalmente coincide con la stazione
- **Observation offering** è un gruppo di osservazioni che sono fornite insieme
  - Singolo sensore
  - Stazione
  - Rete
  - ...

SOS – compatibilità con rmap

- Una **procedure** può coincidere con un sensore
  - e.g. `-/1212345,4312345/rmap/254,0,0/103,2000,-,-/B12101`
- Una **observed property** può coincidere con
  - La terna *(timerange, livello, var)*
    - e.g. `254,0,0/103,2000,-,-/B12101`
  - La sola var
    - e.g. `B12101`
- Una **feature of interest** può coincidere con la stazione
  - e.g. `-/1212345,4312345/rmap`

SOS – implementazione in rmap

- http://rmap.cc/sos?service=SOS&acceptVersions=1.0.0&request=GetCapabilities
- http://rmap.cc/sos/?service=SOS&version=1.0.0&request=DescribeSensor&procedure=urn:rmap:procedure:digiteco/1162336,4465346/rmap/254,0,0/103,2000,-,-/B12101
- http://rmap.cc/sos/?service=SOS&version=1.0.0&request=GetObservation&responseFormat=text/xml;subtype="om/1.0.0"&offering=urn:rmap:procedure:digiteco/1162336,4465346/rmap/254,0,0/103,2000,-,-/B12101&observedProperty=urn:rmap:procedure:digiteco/1162336,4465346/rmap/254,0,0/103,2000,-,-/B12101