

# Architetture del software e dei dati

## Appello 25/02/2015

Francesco Mazzei	748118
Stefano Saldarini	748101
Matteo Zuccon	756417

# Indice

- Introduzione
- Architettura del software
- Architettura dei dati
- Considerazioni

# Introduzione

# Analisi del problema

## Abstract

Realizzazione di un sistema per  
l'osservazione della situazione  
idrogeologica del territorio e per la  
segnalazione di emergenze

# Analisi del problema

## Studio del problema

Il sistema deve supportare:

- L'acquisizione in tempo reale di dati idrometrici (livello dei corsi d'acqua) attraverso opportuni sensori
- L'acquisizione di segnalazioni di emergenze gravi
- L'identificazione di situazioni di emergenza potenziali a medio termine (alcune ore), attraverso l'incrocio delle informazioni meteo e i dati idrometrici

# Analisi del problema

## Studio del problema

- La pianificazione degli spostamenti delle squadre di emergenza in base alle informazioni relative alle emergenze potenziali
- La notifica della pianificazione ai responsabili territoriali della protezione civile e alle squadre di emergenza coinvolte
- La memorizzazione della pianificazione degli spostamenti delle squadre di emergenza
- La notifica di emergenze gravi alle squadre di emergenza più prossime

# Analisi del problema

## Acronimi

- DI - dato idrometrico
- SEP - segnalazione emergenza potenziale
- SEG - segnalazione emergenza grave
- BDM - base dati meteo (esterna)
- BSE - base dati segnalazioni emergenza
- BRI - base dati rete idrica

# Analisi del problema

## Ambiguità

- Qual è il target del sistema? (regionale? nazionale?)
- Chi sono gli operatori a campo che segnalano SEG?
- Come vengono notificate le pianificazioni degli spostamenti?
- Come vengono resi visibili le informazioni SEP?
- Differenza fra informazioni SEP e informazioni SEP sintetiche?



# Analisi del problema

## Ambiguità

- L'algoritmo che identifica una SEP sulla base di quanti dati lavora?
- Come interagiamo con BDM? Ogni quanto vengono aggiornate le previsioni?
- Chi pianifica gli spostamenti delle squadre in base a SEP?
- Come identifico le squadre di emergenza più prossime?

# Analisi del problema

## Assunzioni

- Il target del sistema è il territorio nazionale italiano
- Presenza di 2000 sensori idrici dislocati sul territorio nazionale (a febbraio 2015 in Lombardia sono presenti circa 100 sensori – Arpa Lombardia)
- I sensori idrici inviano i dati rilevati ogni ora (tutti i sensori sono sincronizzati – l'intervallo di rilevazione è analogo a quello di sistemi realmente esistenti)

# Analisi del problema

## Assunzioni

- I sensori sono programmabili (end-point invio dati e frequenza invio)
- La trasmissione di DI avviene tramite sensori idrometrici dotati di modulo GPRS (con output digitale)
- La pianificazione degli spostamenti delle squadre di emergenza **non** viene svolta in automatico dal sistema, ma da un operatore del centro di supervisione

# Analisi del problema

## Assunzioni

- I nodi idrici si trovano in corrispondenza del punto di confluenza di due corsi d'acqua
- Il tratto di fiume va da un nodo al successivo
- Vengono monitorati solo i tratti d'acqua considerati a rischio

# Analisi del problema

## Assunzioni

- Una squadra di emergenza può essere sul campo o nella sede operativa
- L'identificazione della squadra di emergenza più prossima avviene semplicemente considerando la distanza fra il luogo in cui avviene l'emergenza e la sede operativa della squadra (solamente se la squadra non è impegnata sul campo)

# Analisi del problema

## Assunzioni

- Il campo regione della tabella *nodo d'acqua* in BRI e il campo denominazione della tabella *regione* in BDM hanno lo stesso dominio
- Conosciamo lo schema logico di BDM
- L'*id* della tabella BDM.previsioniMeteo e l'*id* della tabella BSE.previsioni identificano la stessa previsione

# Analisi del problema

## Stime

- Esistono circa 1000 fiumi in Italia (non tutti necessitano monitoraggio)
- L'altezza massima registrata è di circa 9 metri (Po affluenza con il Ticino)

# Analisi del problema

## Vincoli funzionali

- Un operatore a campo può segnalare una SEG
- Il sistema identifica automaticamente una SEP
- Il sistema permette all'operatore del centro di supervisione di pianificare gli spostamenti delle squadre di emergenza per gestire la SEP
- La pianificazione deve essere notificata ai responsabili territoriali della protezione civile e alle squadre di emergenza coinvolte
- La pianificazione deve essere memorizzata su BSE



# Analisi del problema

## Vincoli funzionali

- Le informazioni di SEP devono essere rese visibili in forma dettagliata agli operatori di un centro di supervisione e ai responsabili territoriali della protezione civile
- Le informazioni di SEP devono essere rese visibili in forma sintetica alla popolazione interessata
- Le informazioni SEG devono essere notificate alle squadre di emergenza più prossime

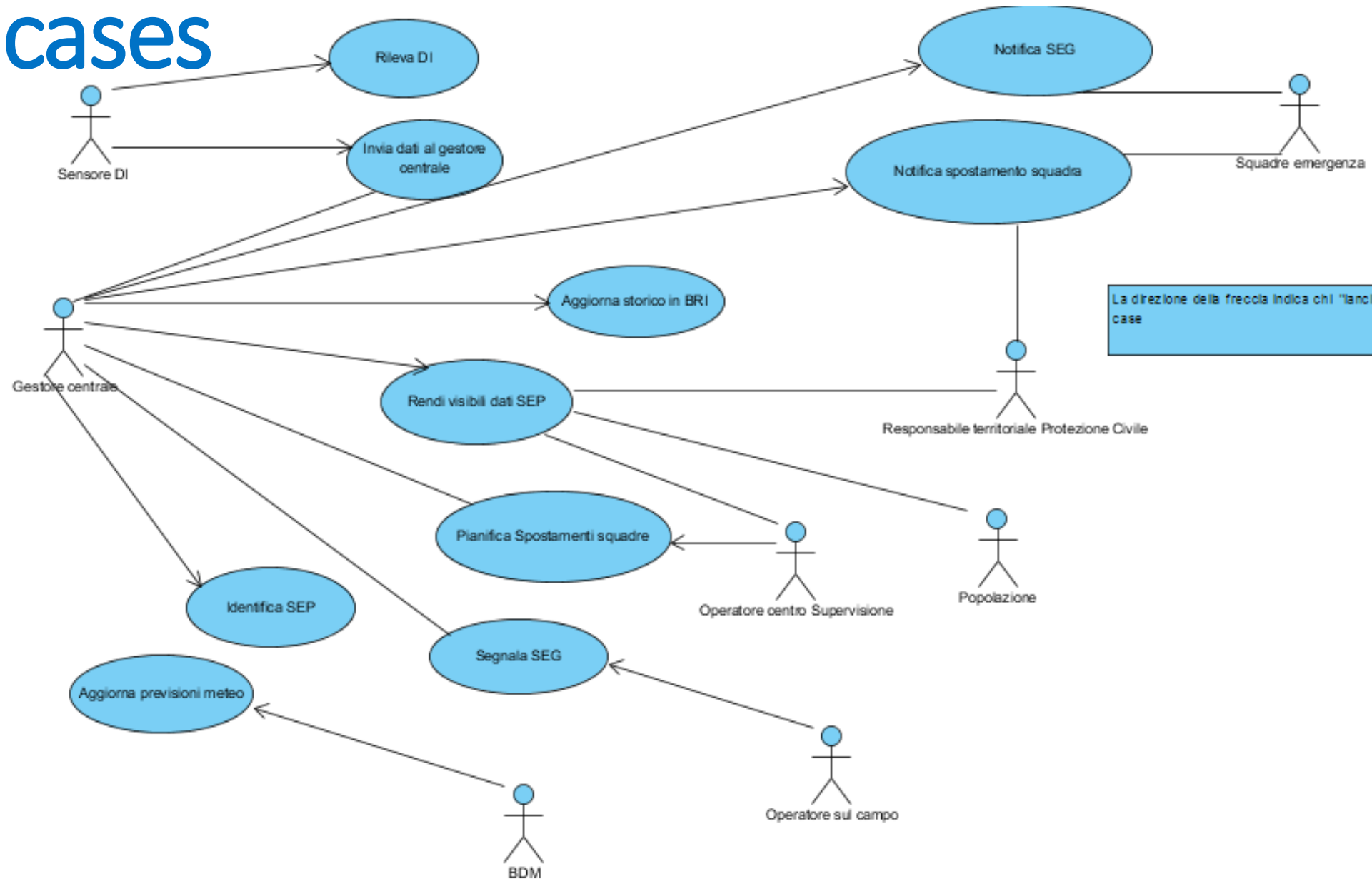
# Analisi del problema

## Vincoli non funzionali

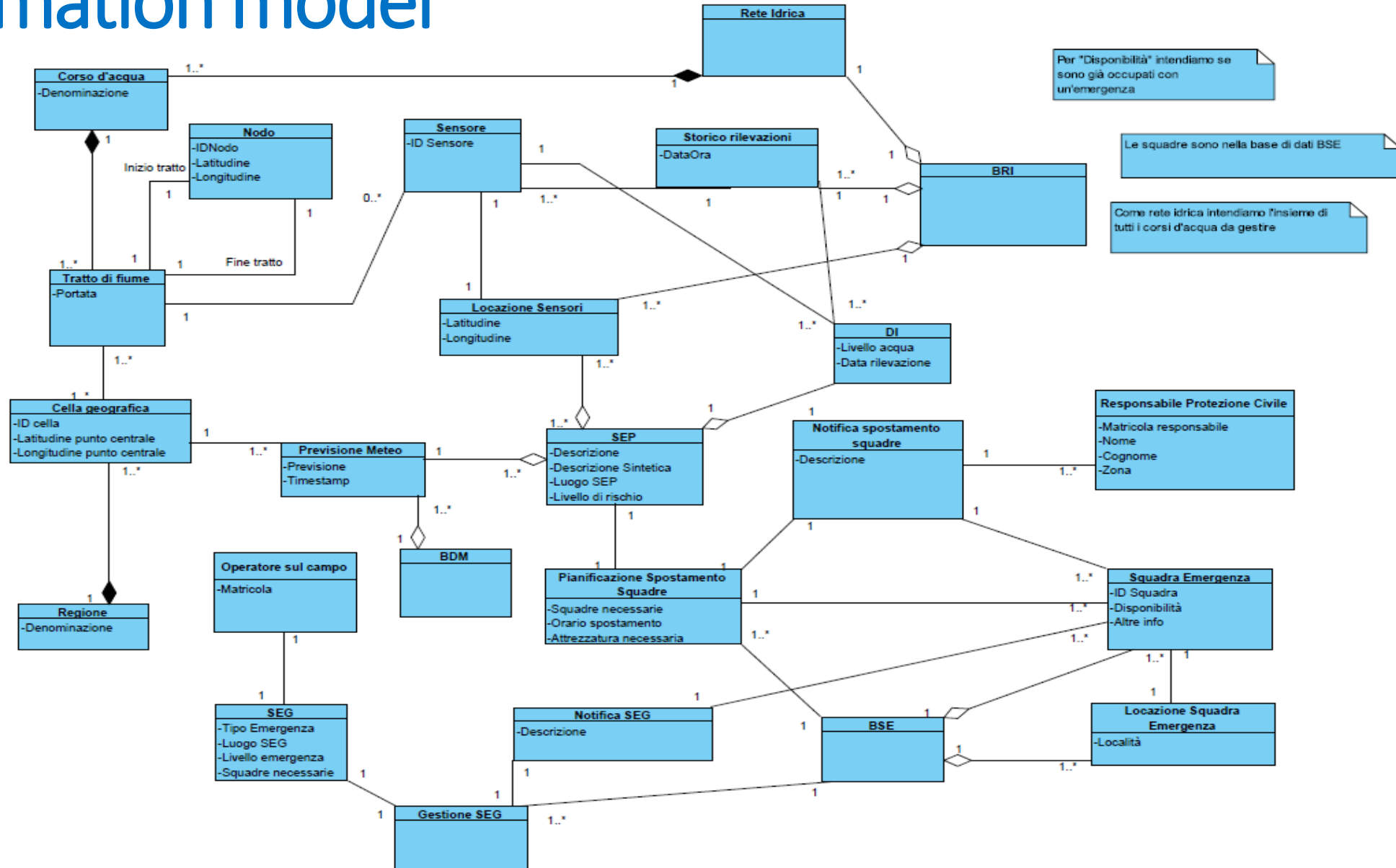
- La ricezione di DI deve essere in *real-time* (1 rilevazione all'ora per ogni sensore)
- Il sistema deve essere disponibile h24
- Tempestività notifica SEG (tempistica non specificata nel testo)

# Architettura del Software

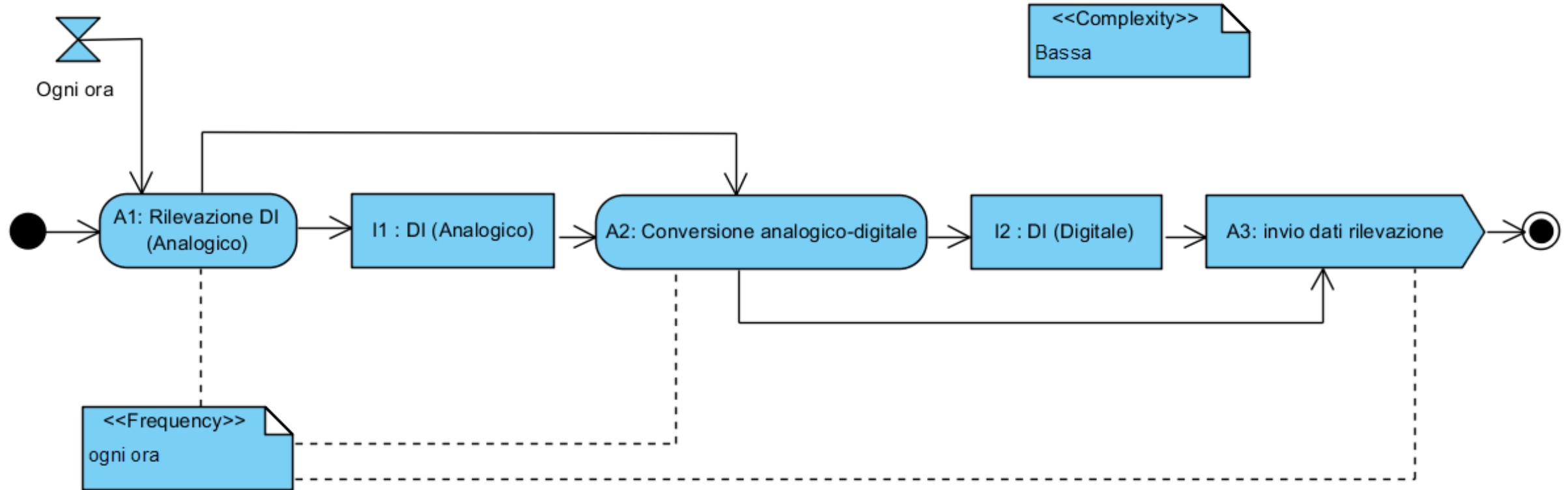
# Use cases



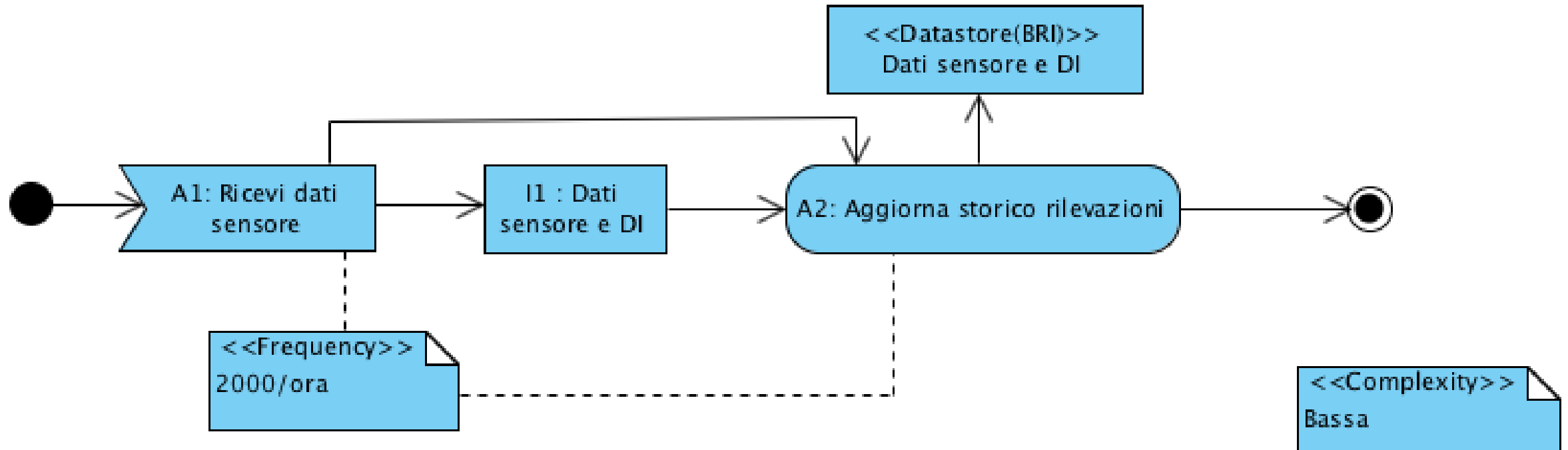
# Information model



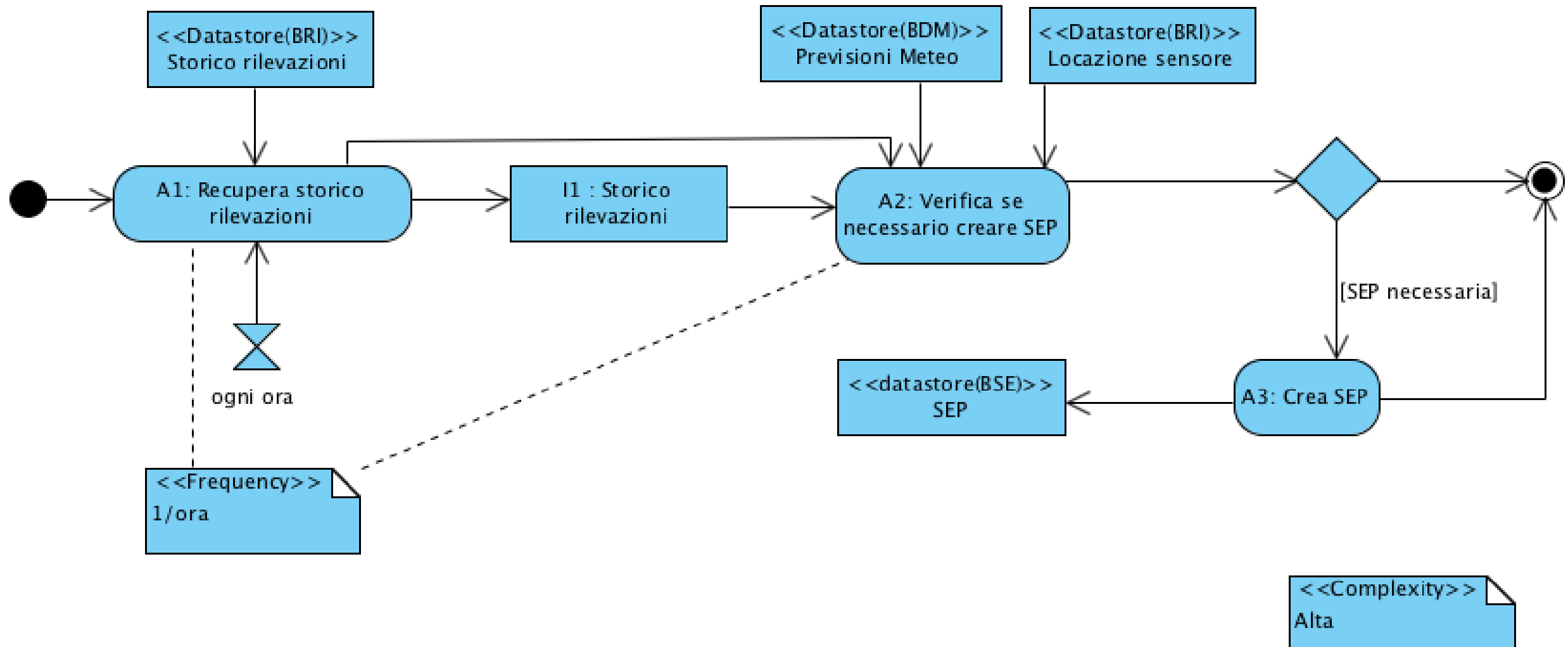
# Information flow – Rileva DI



# Information flow – Aggiorna storico

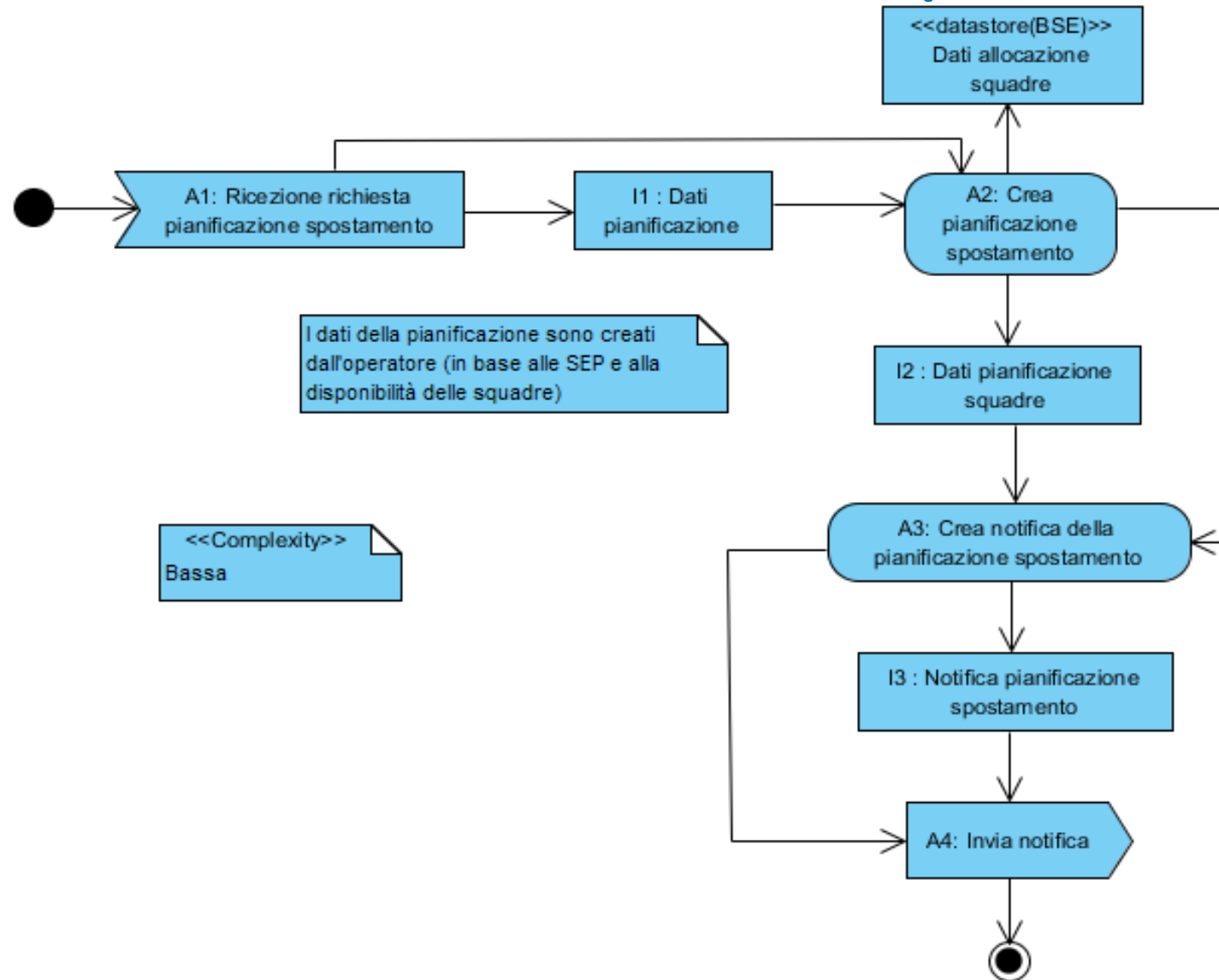


# Information flow – Identifica SEP

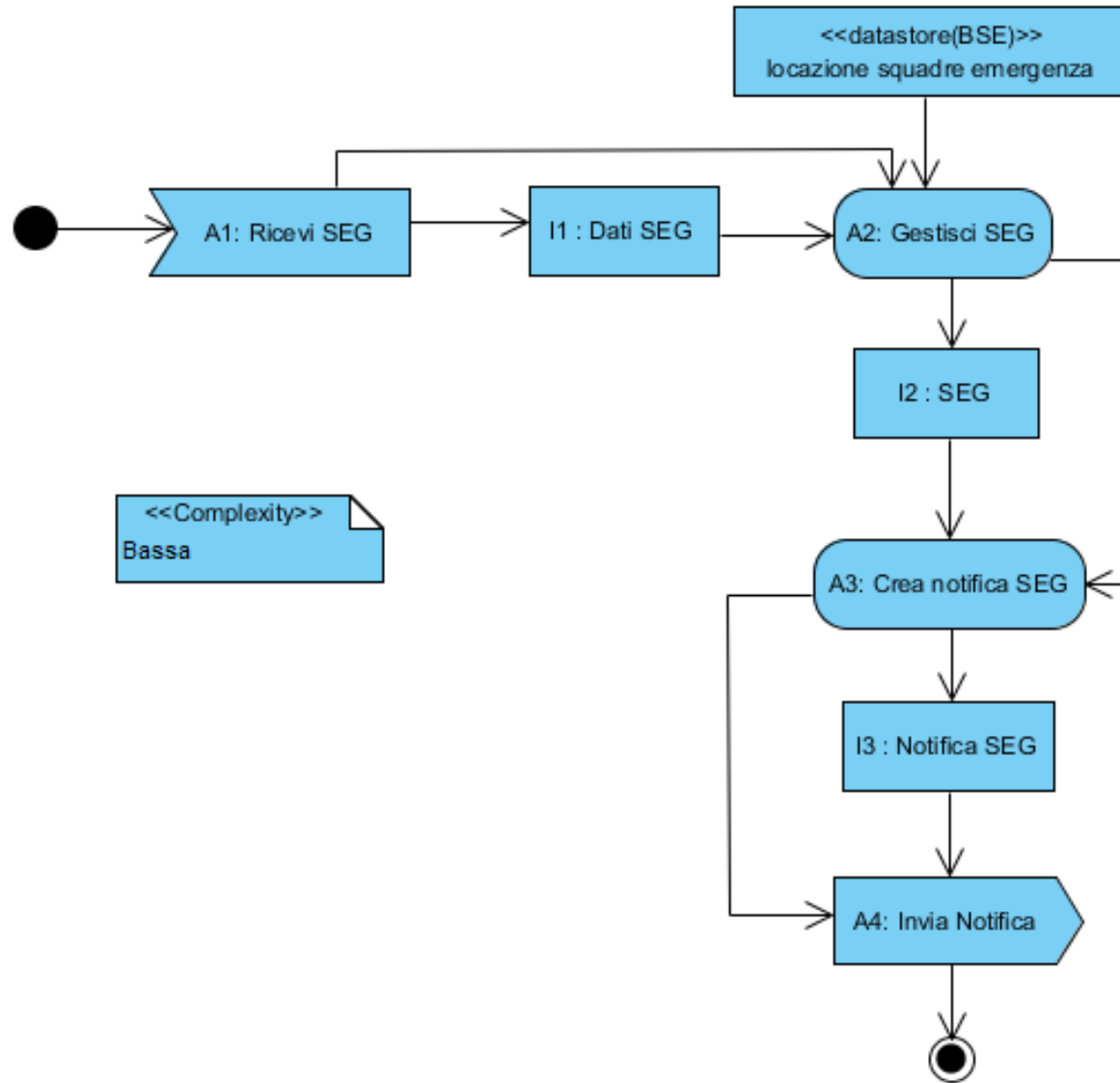




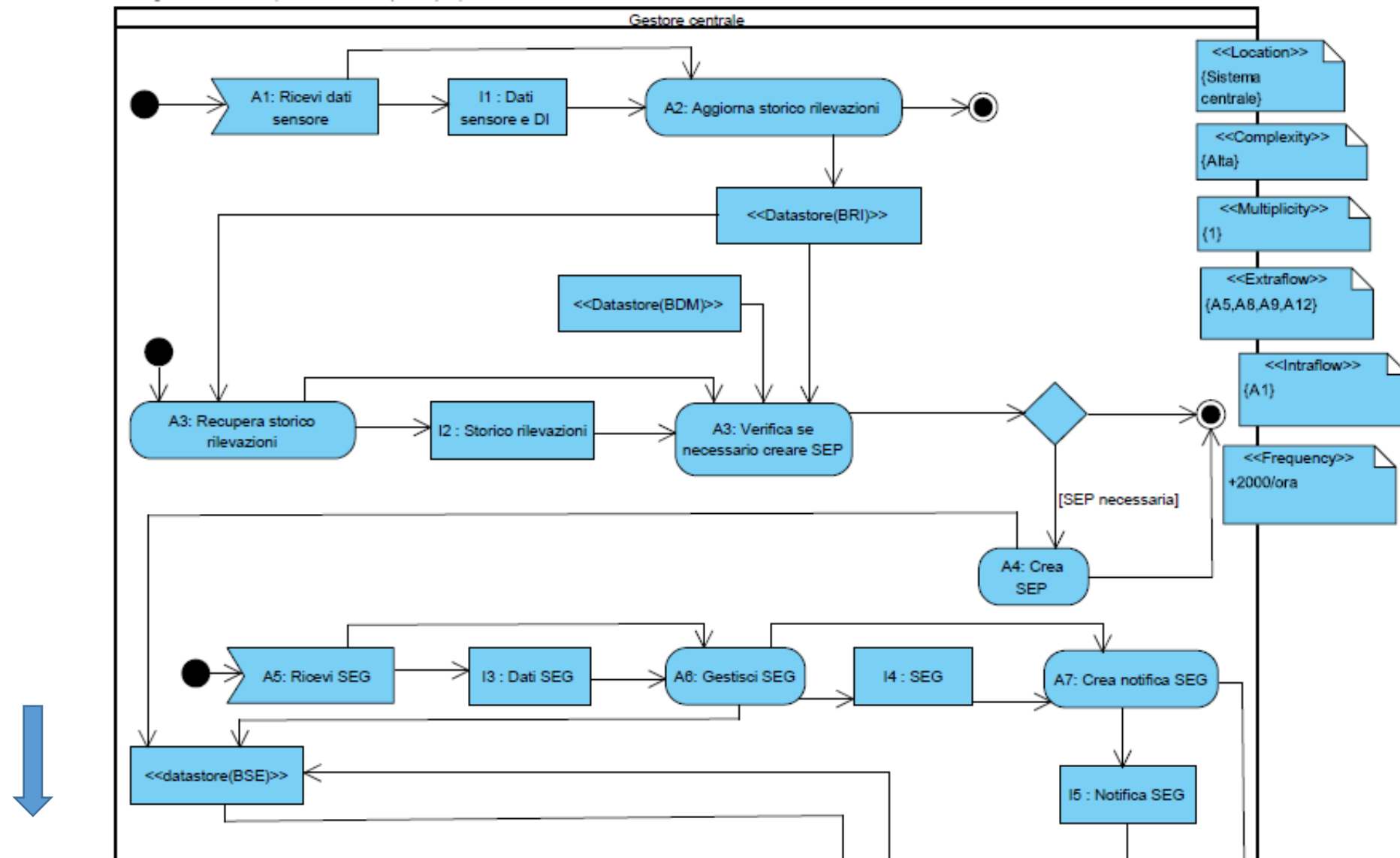
# Information flow – Pianifica spostamento squadre e.



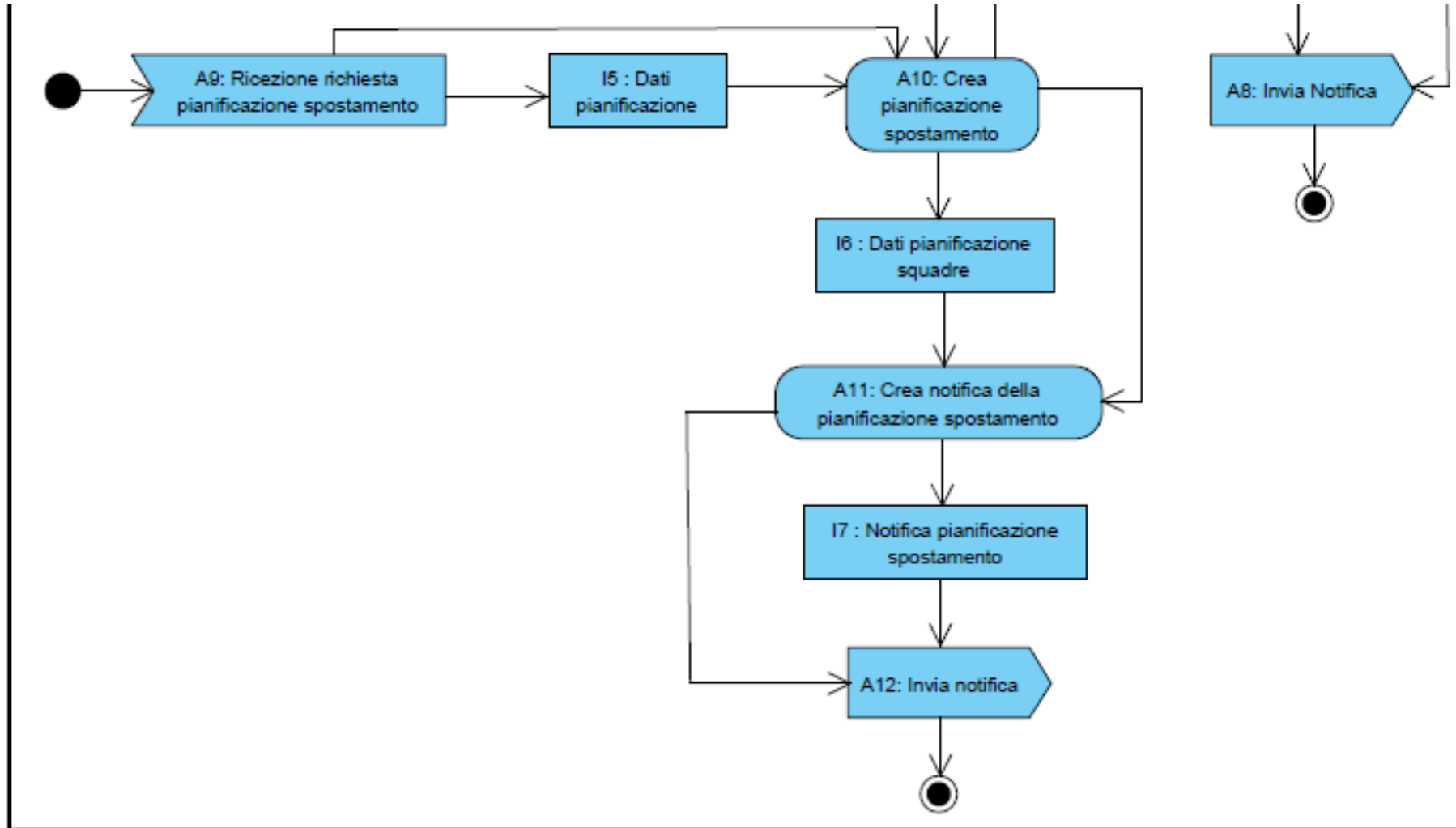
# Information flow – Ricevi SEG



# Componenti logici – soluzione 1 - Gestore Centrale



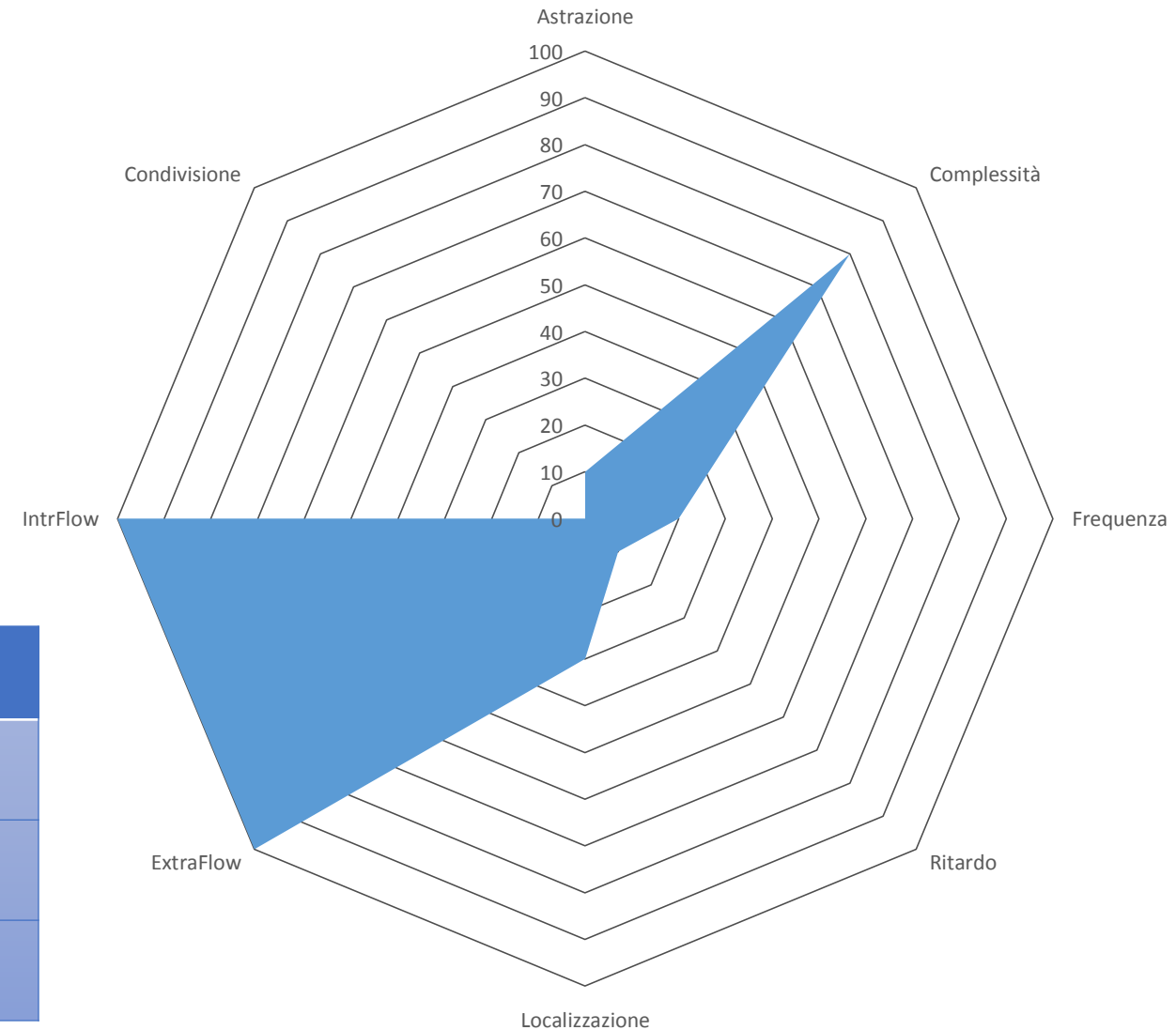
# Componenti logici – soluzione 1 - Gestore Centrale



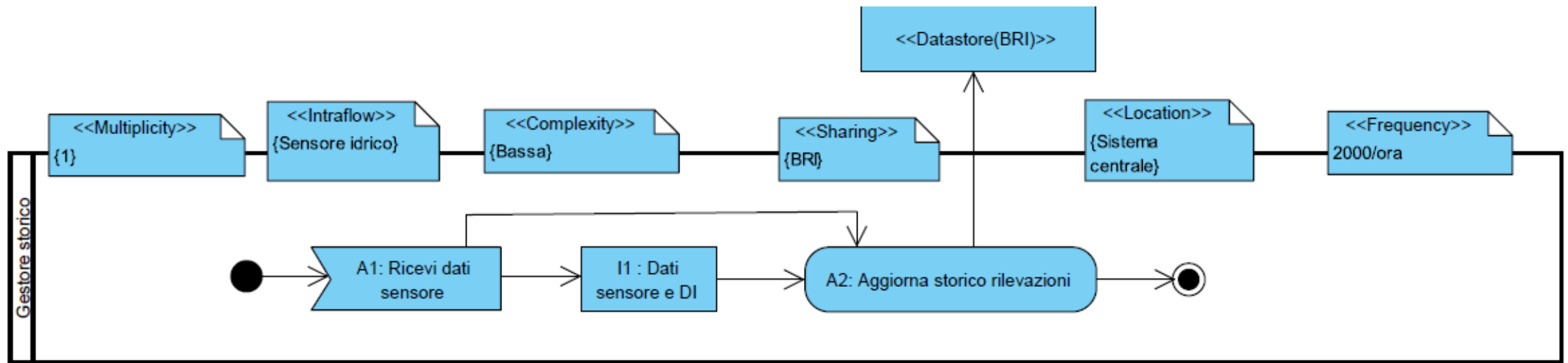
# Componenti logici – soluzione 1

Astrazione	10
Complessità	80
Frequenza	20
Ritardo	10
Localizzazione	30
ExtraFlow	100
IntrFlow	100
Condivisione	0

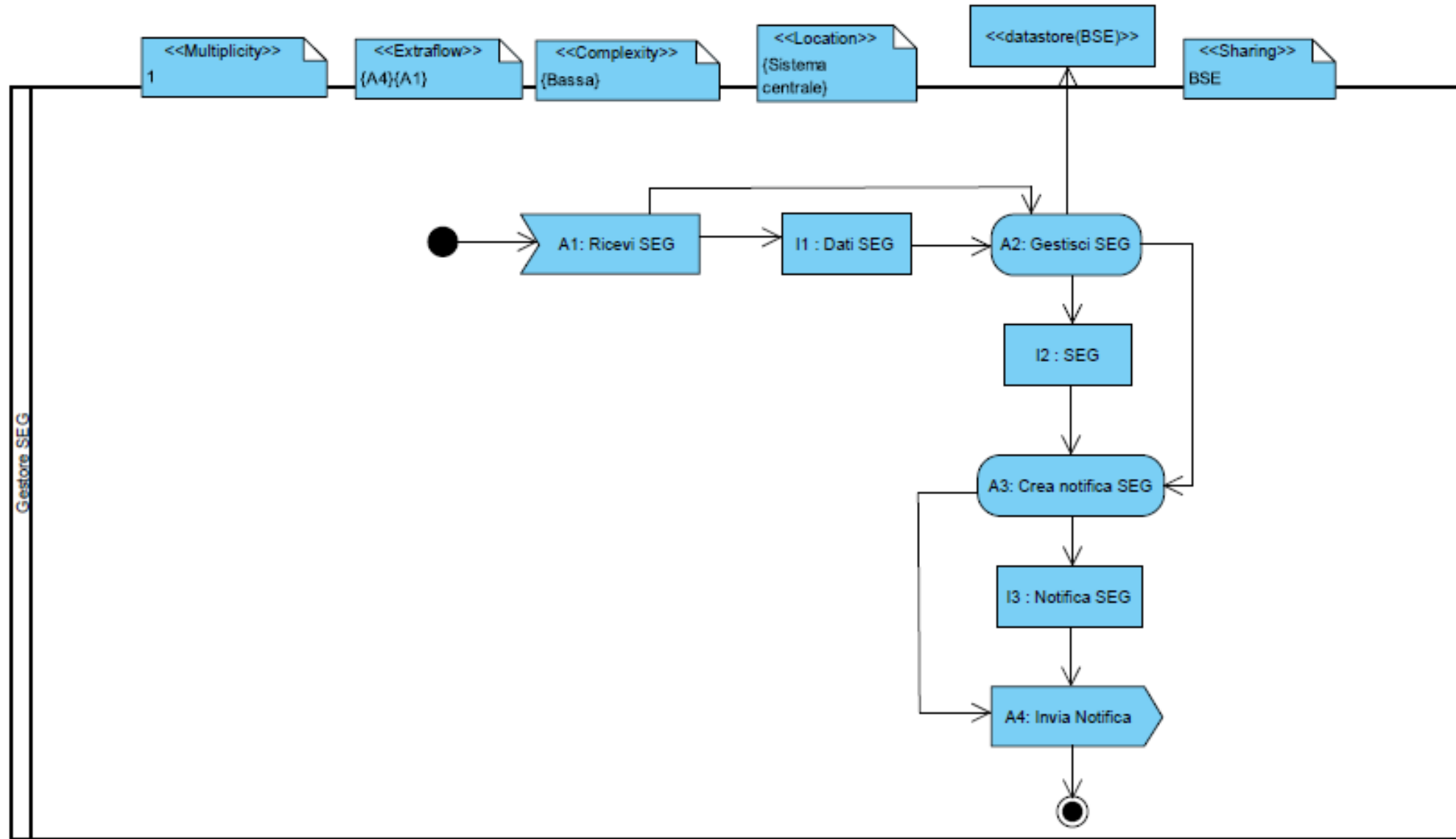
Pro	Contro
Condivisione bassa	ExtraFlow elevato
	IntraFlow elevato
	Complessità elevata



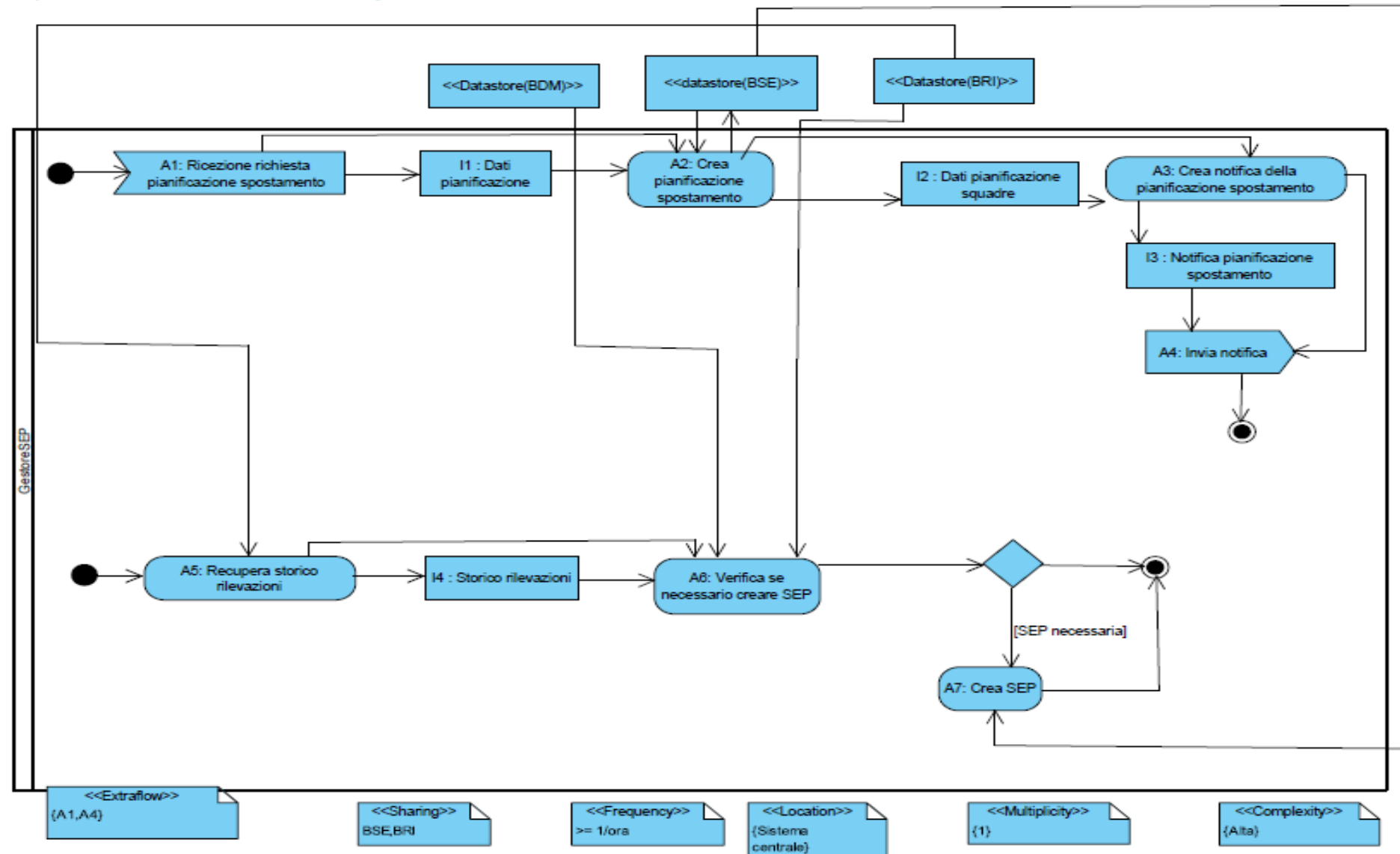
# Componenti logici – soluzione 2 - Gestore storico



# Componenti logici – soluzione 2 - Gestore SEG



# Componenti logici – soluzione 2 - Gestore SEP

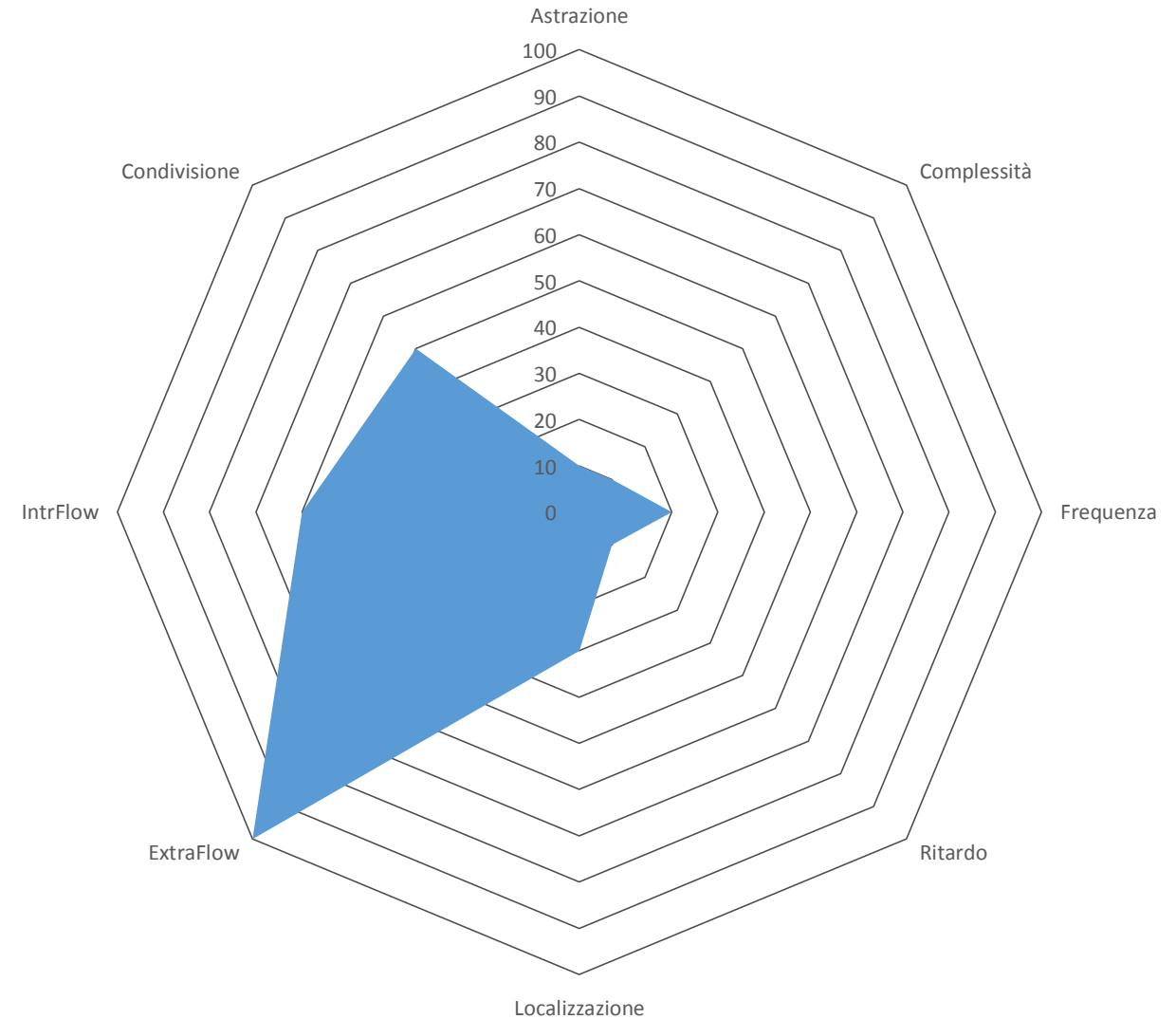




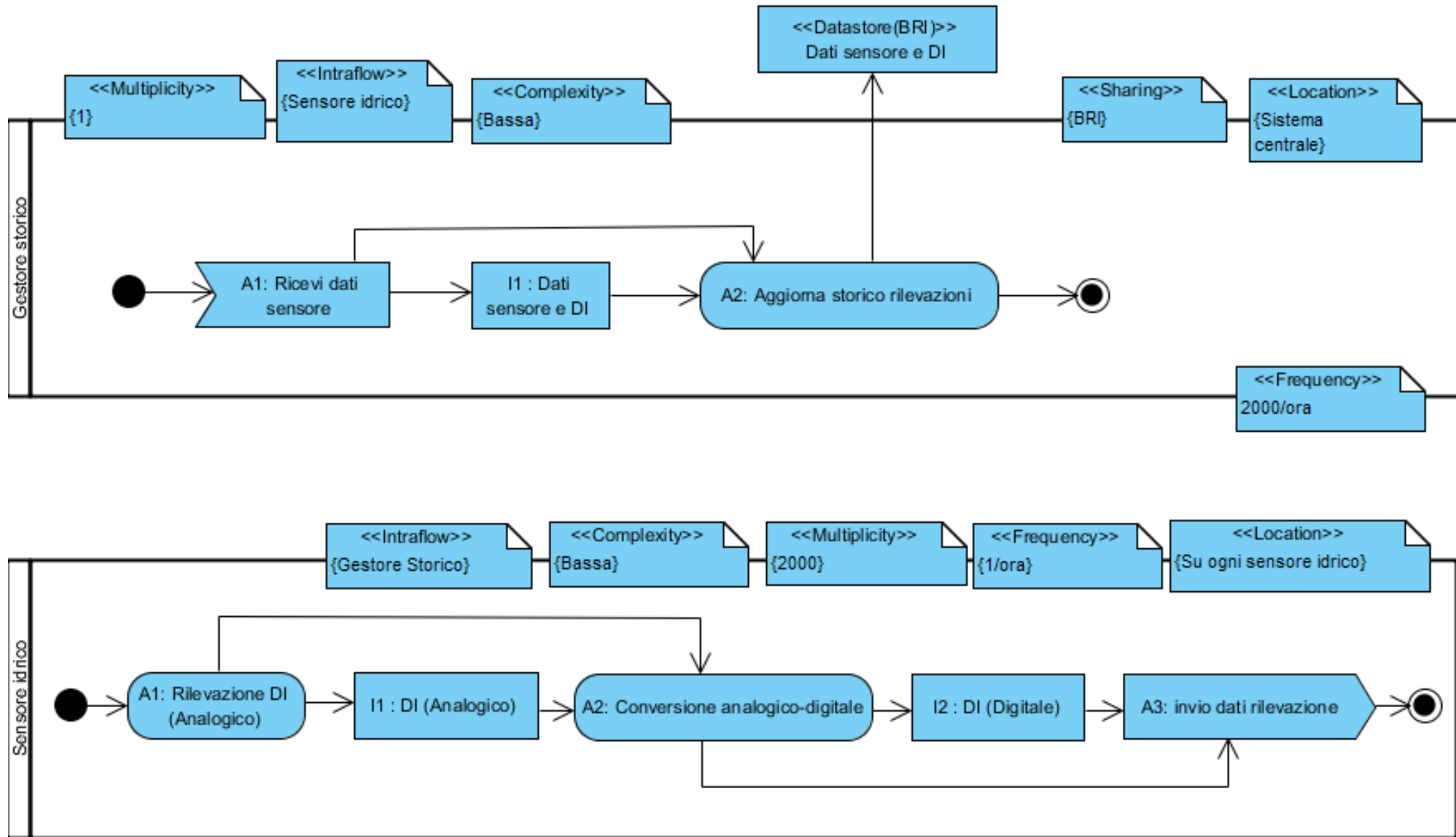
# Componenti logici – soluzione 2

Astrazione	10
Complessità	10
Frequenza	20
Ritardo	10
Localizzazione	30
ExtraFlow	100
IntrFlow	60
Condivisione	50

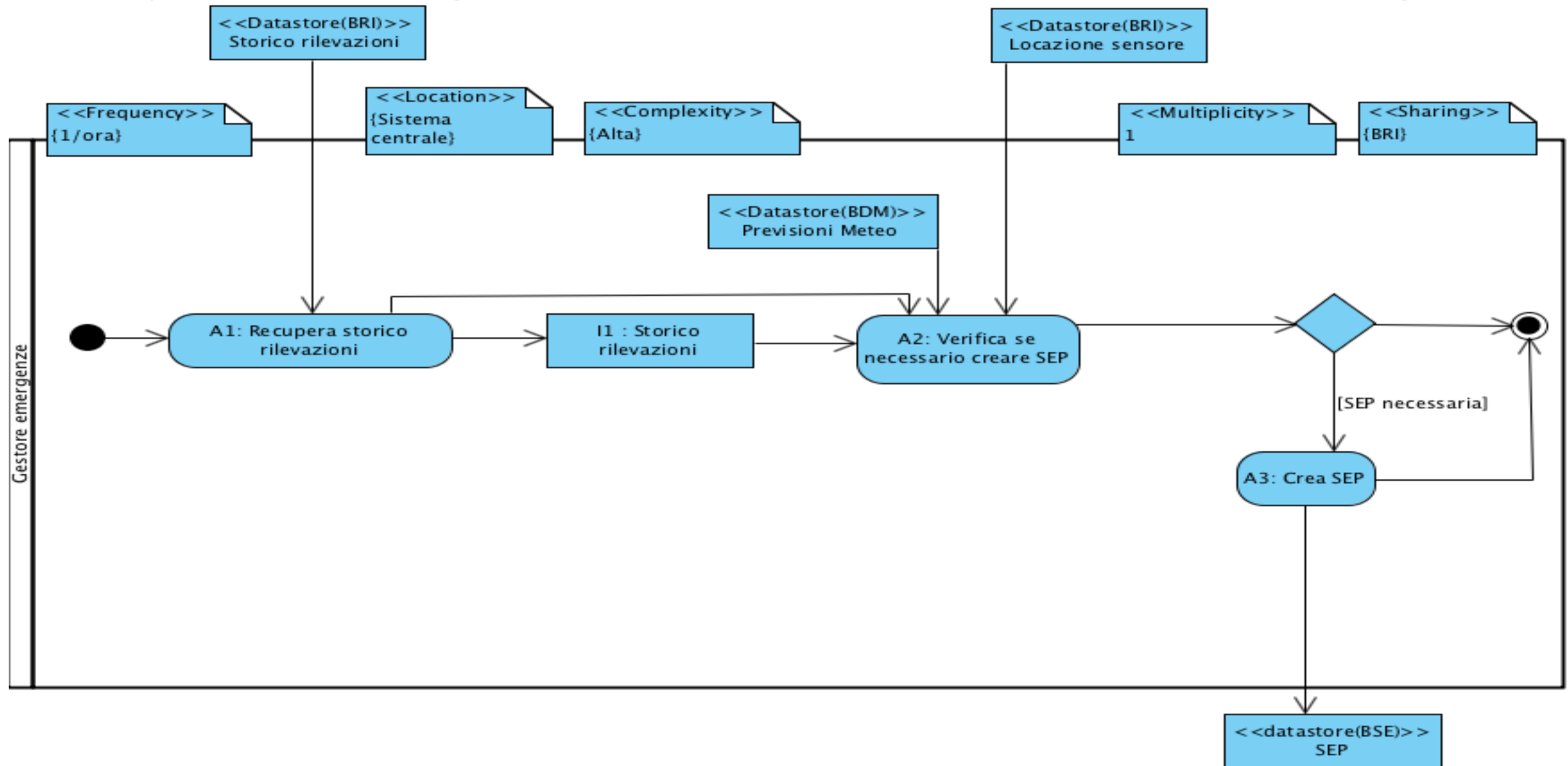
Pro	Contro
Complessità bassa	ExtraFlow elevato
Condivisione media	



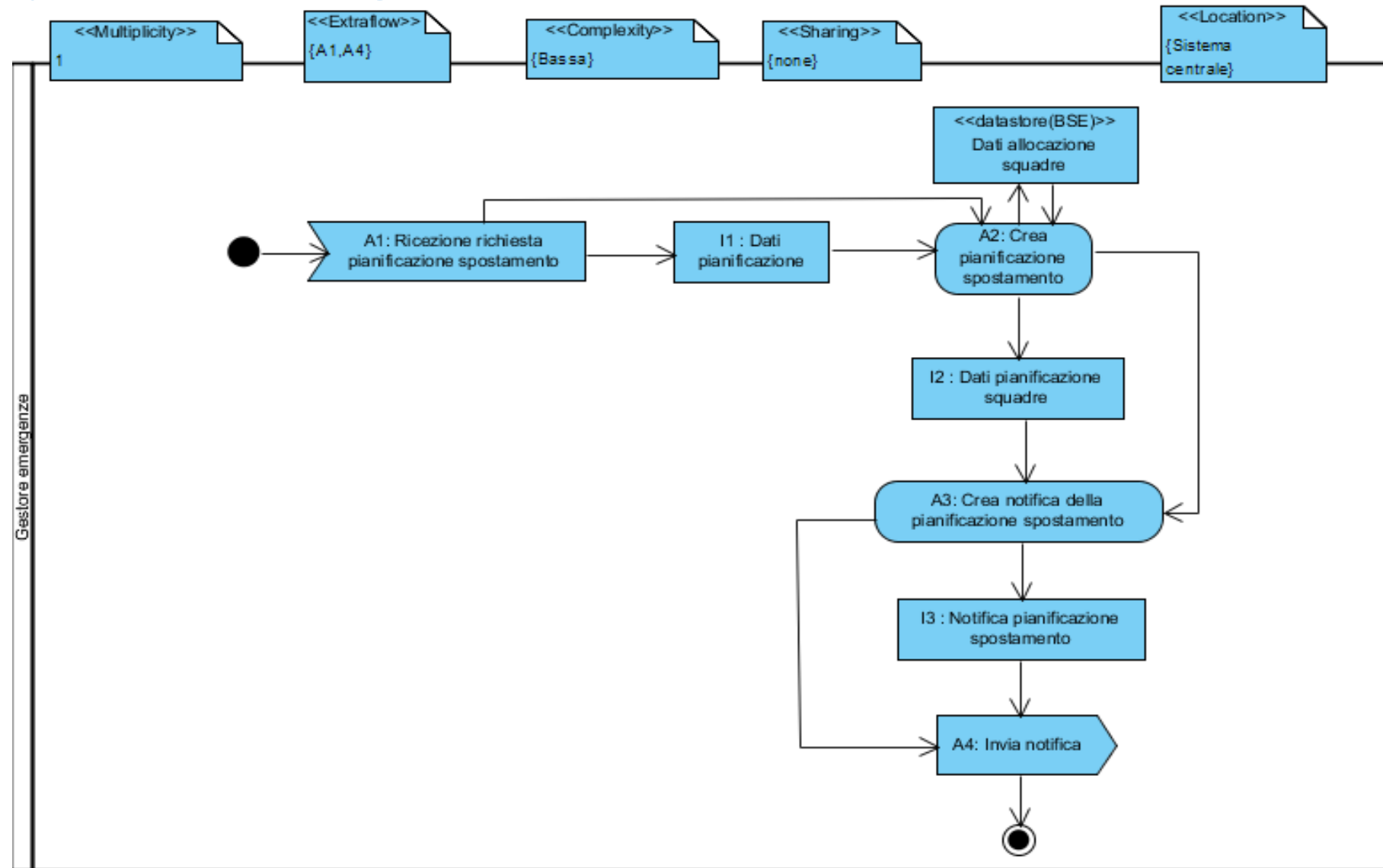
# Componenti logici – soluzione 3 -Gestore storico/Sensore idrico



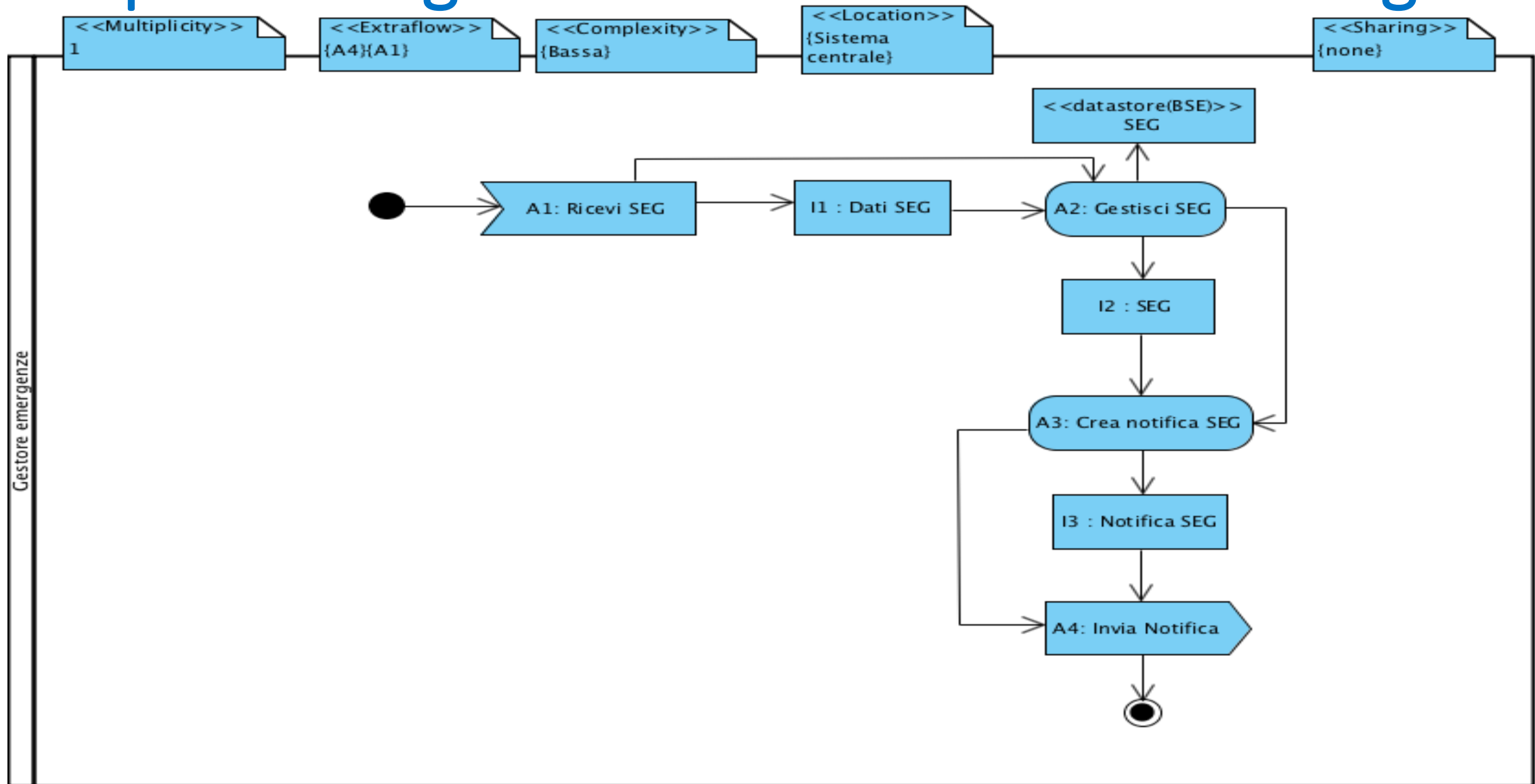
# Componenti logici – soluzione 3 –Gestore emergenze



# Componenti logici – soluzione 3 –Gestore emergenze



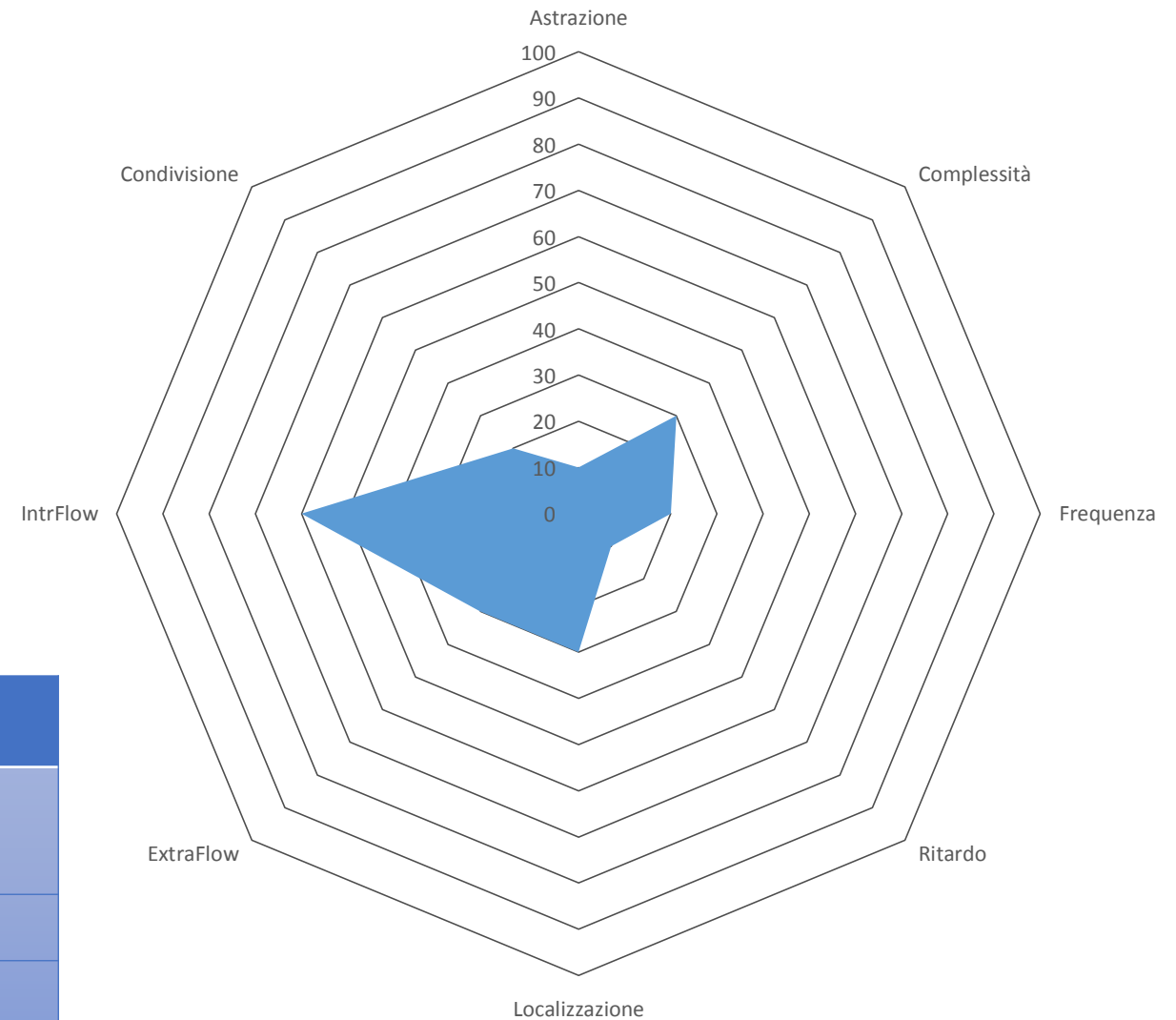
# Componenti logici – soluzione 3 –Gestore emergenze



# Componenti logici – soluzione 3

Astrazione	10
Complessità	30
Frequenza	20
Ritardo	10
Localizzazione	30
ExtraFlow	30
IntrFlow	60
Condivisione	20

Pro	Contro
Valori medio bassi	IntraFlow di medio valore
Extraflow basso	Complessita medio bassa
Condivisione bassa	

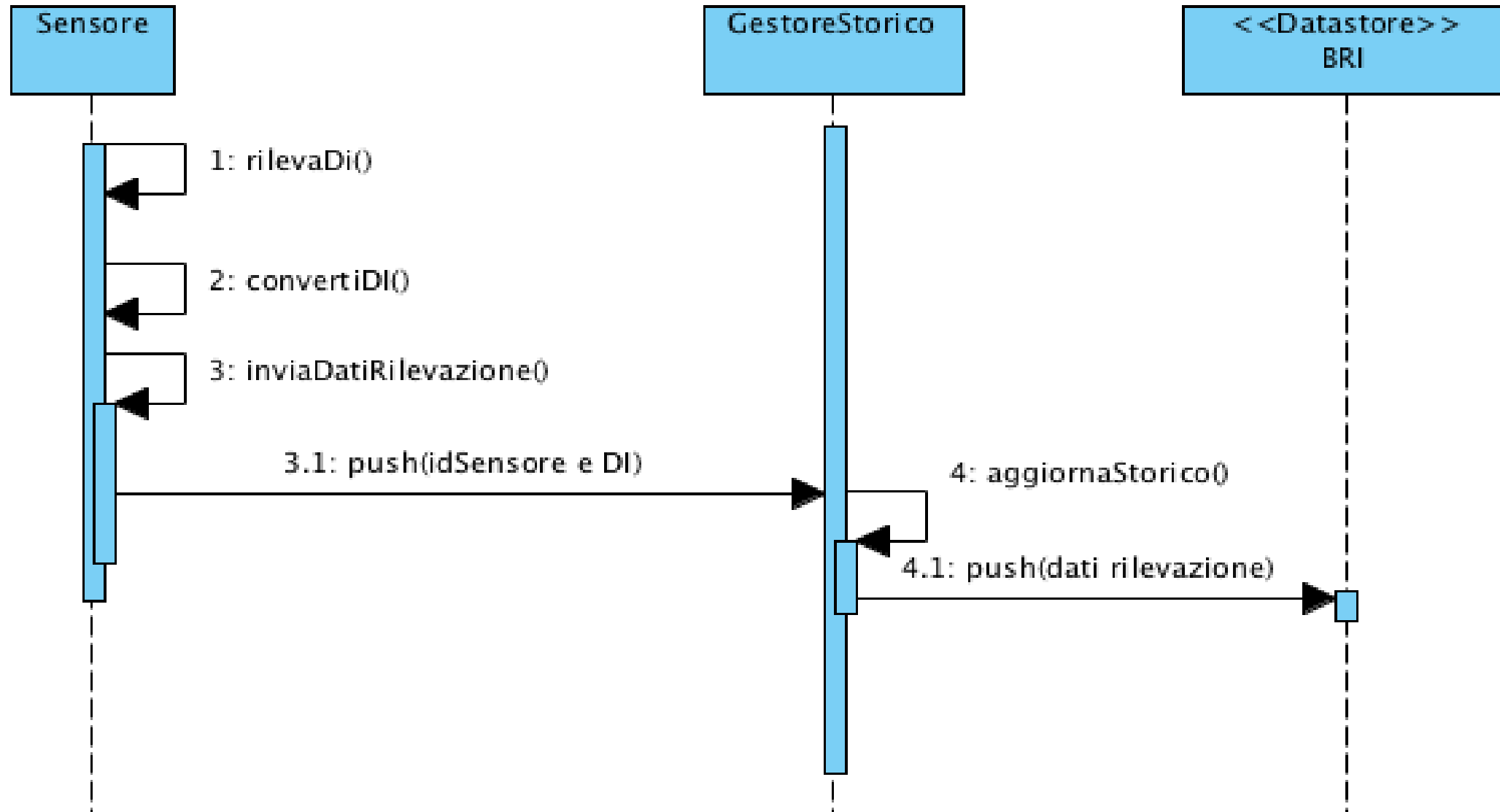


# Componenti logici

Analizzando i pro/contro delle diverse soluzioni, abbiamo scelto la terza soluzione poiché rappresenta i migliori compromessi tra le diverse proprietà analizzate.

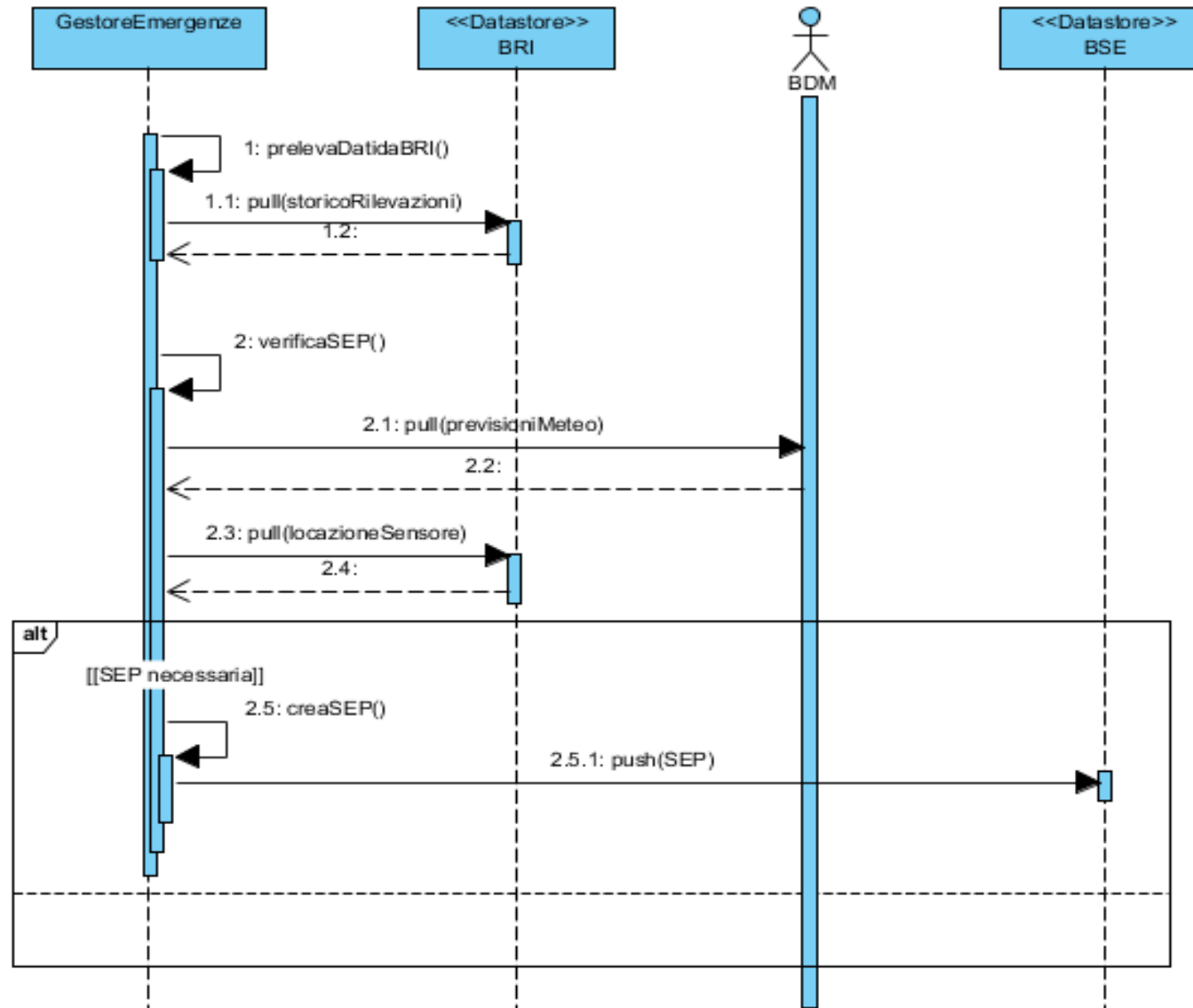


# Sequence diagrams – Aggiorna storico

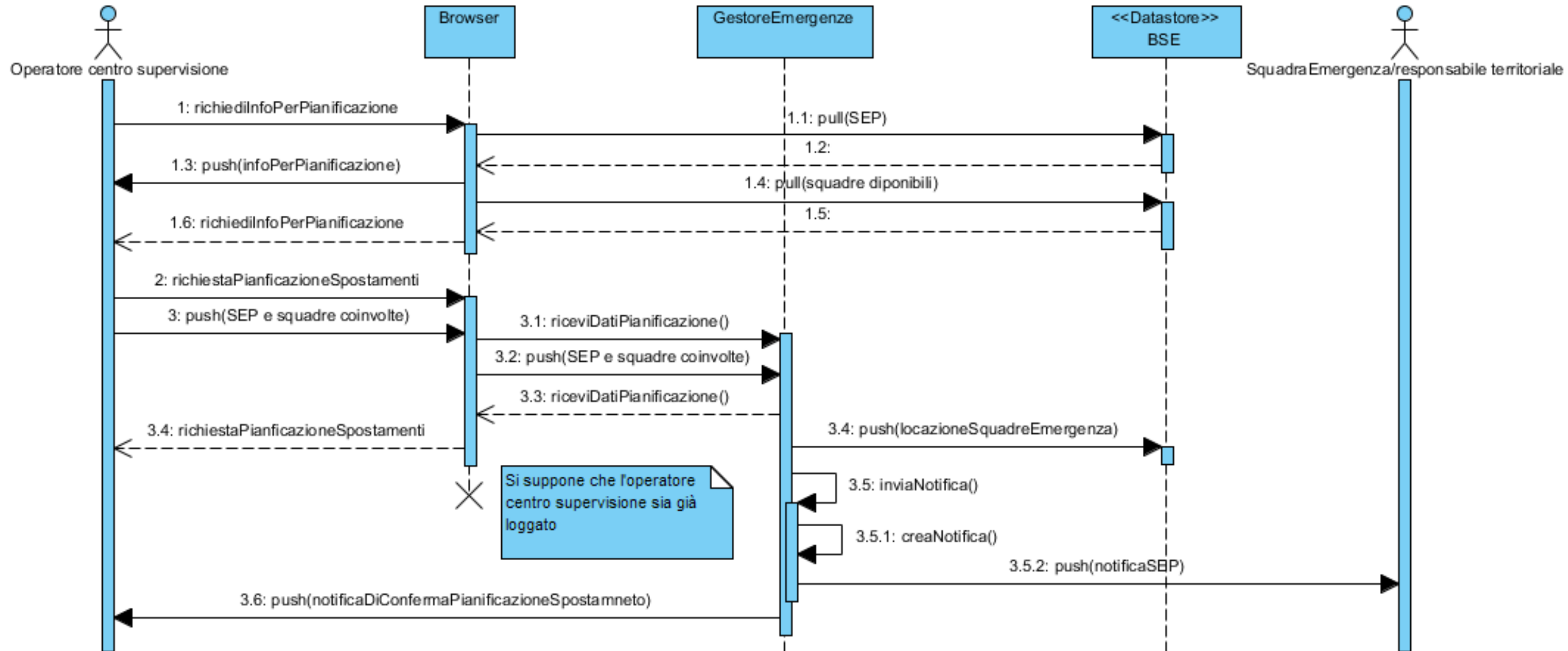




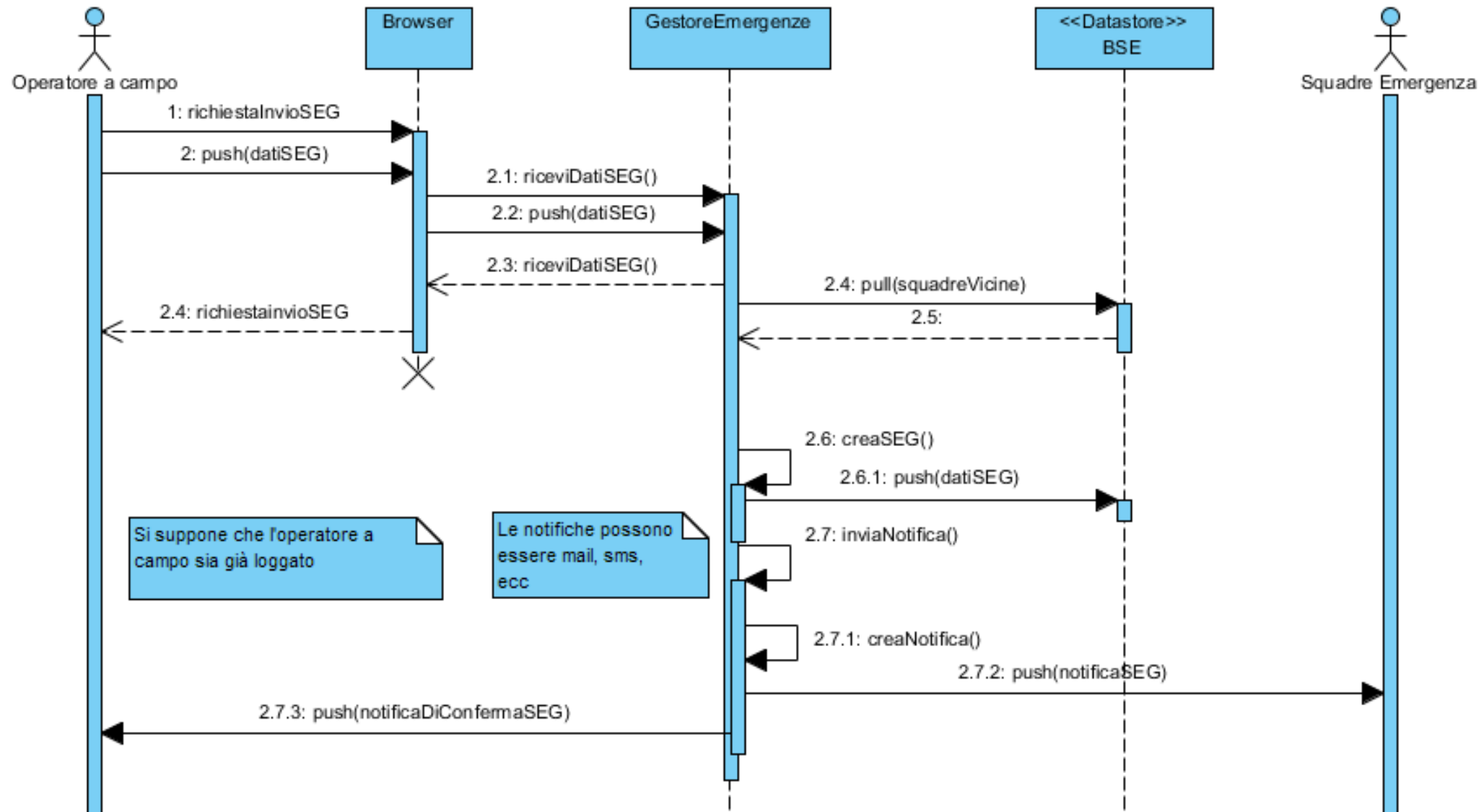
# Sequence diagrams – Identifica SEP



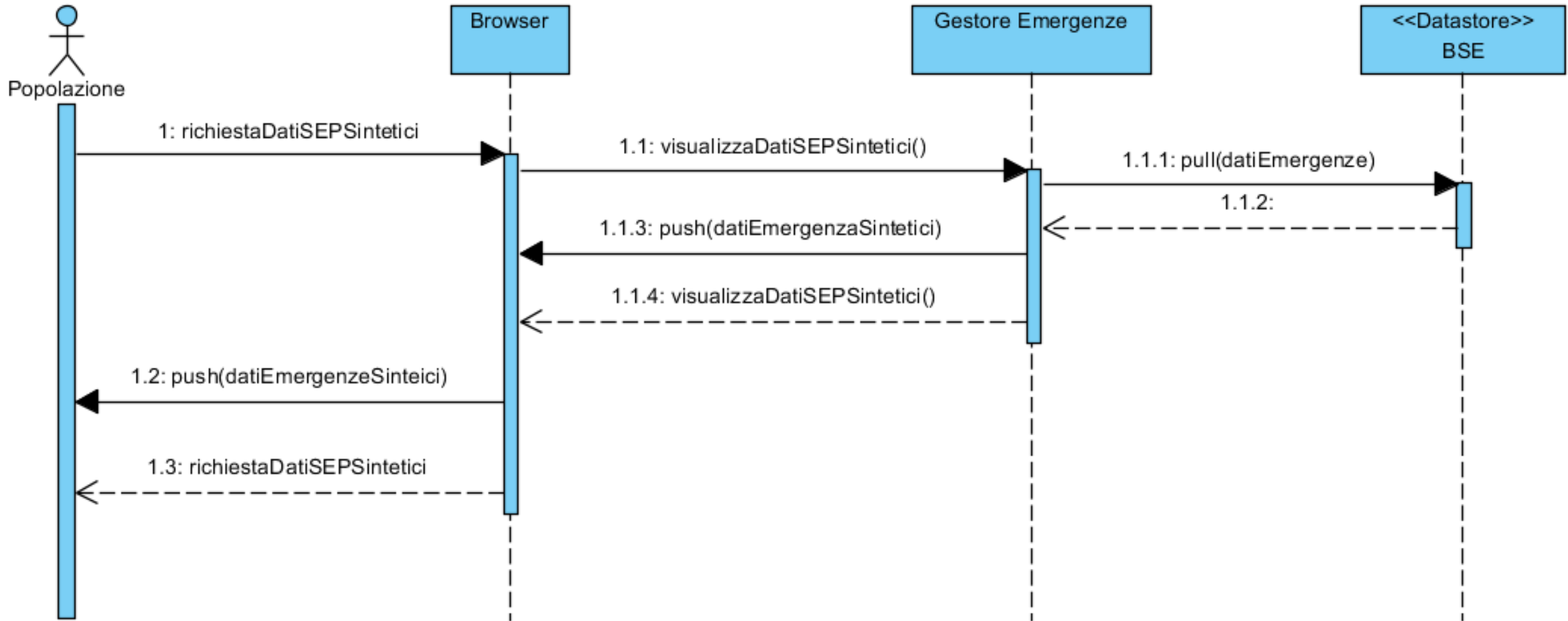
# Sequence diagrams – Pianifica spostamento squadre e.



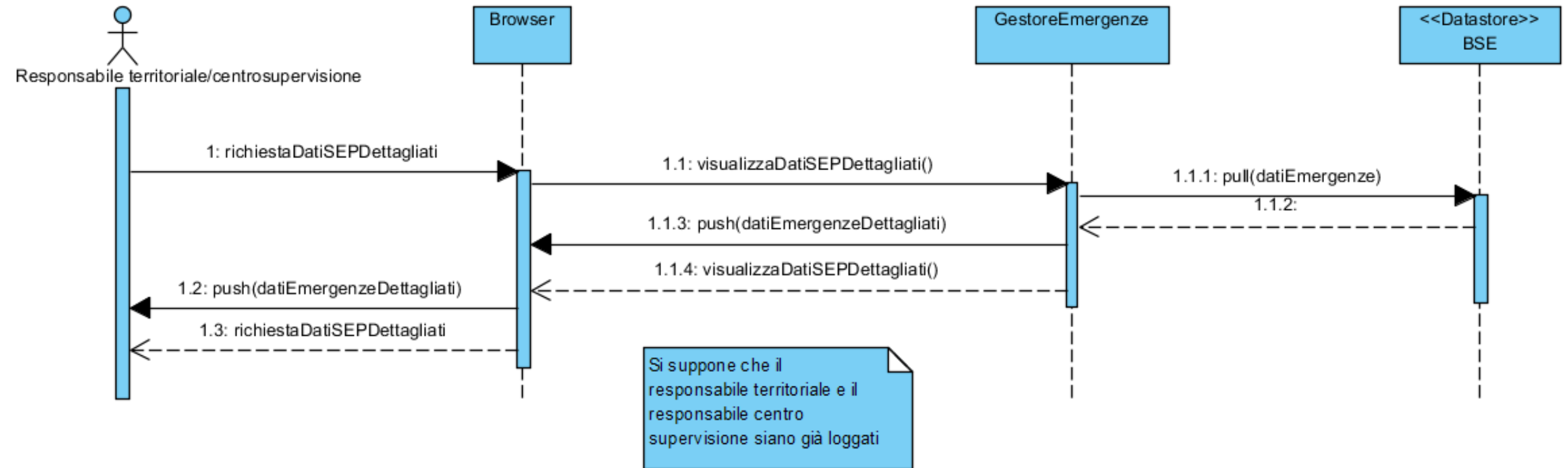
# Sequence diagrams – Gestisci SEG



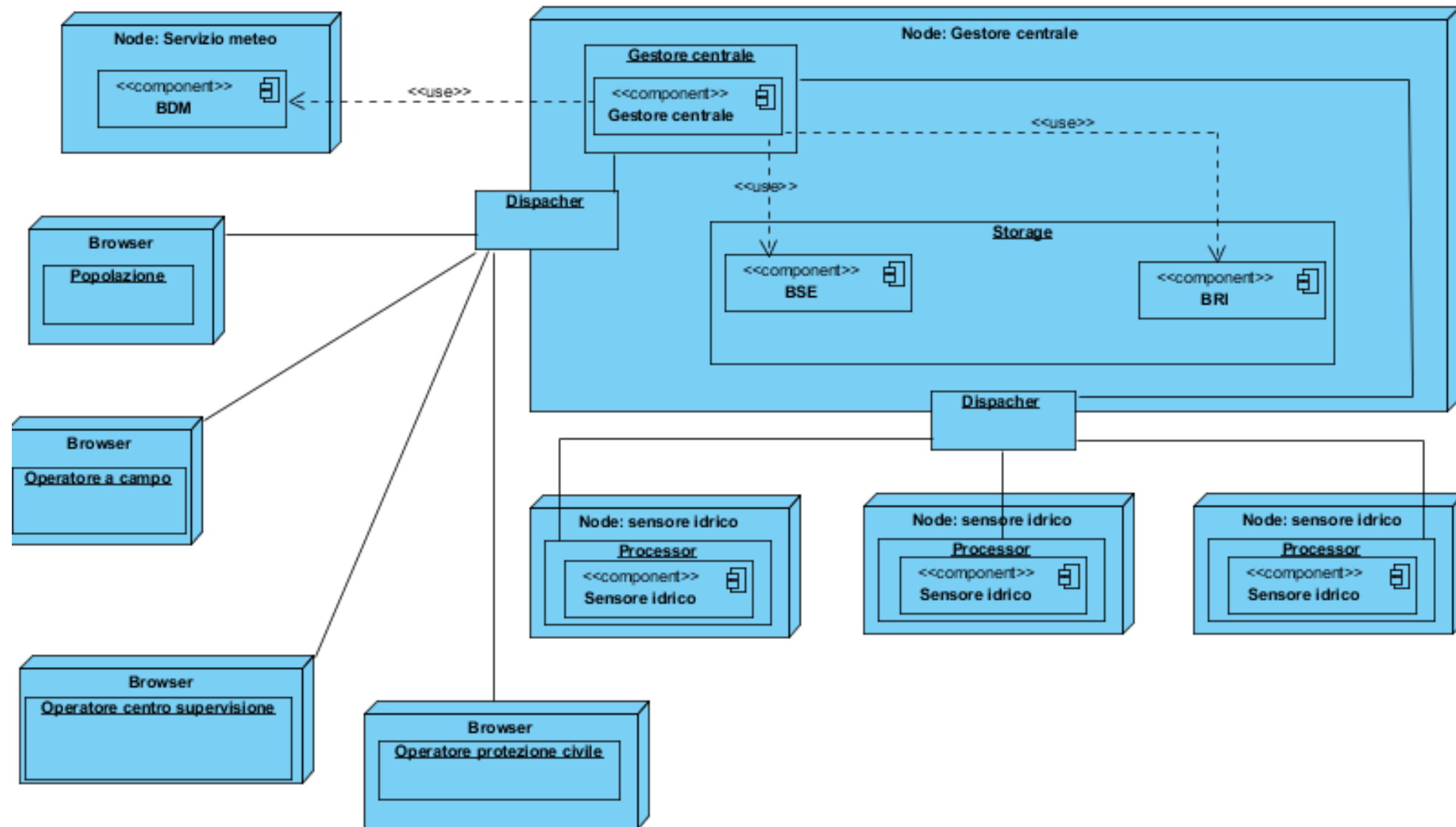
# Sequence diagrams – visualizza dati sintetici



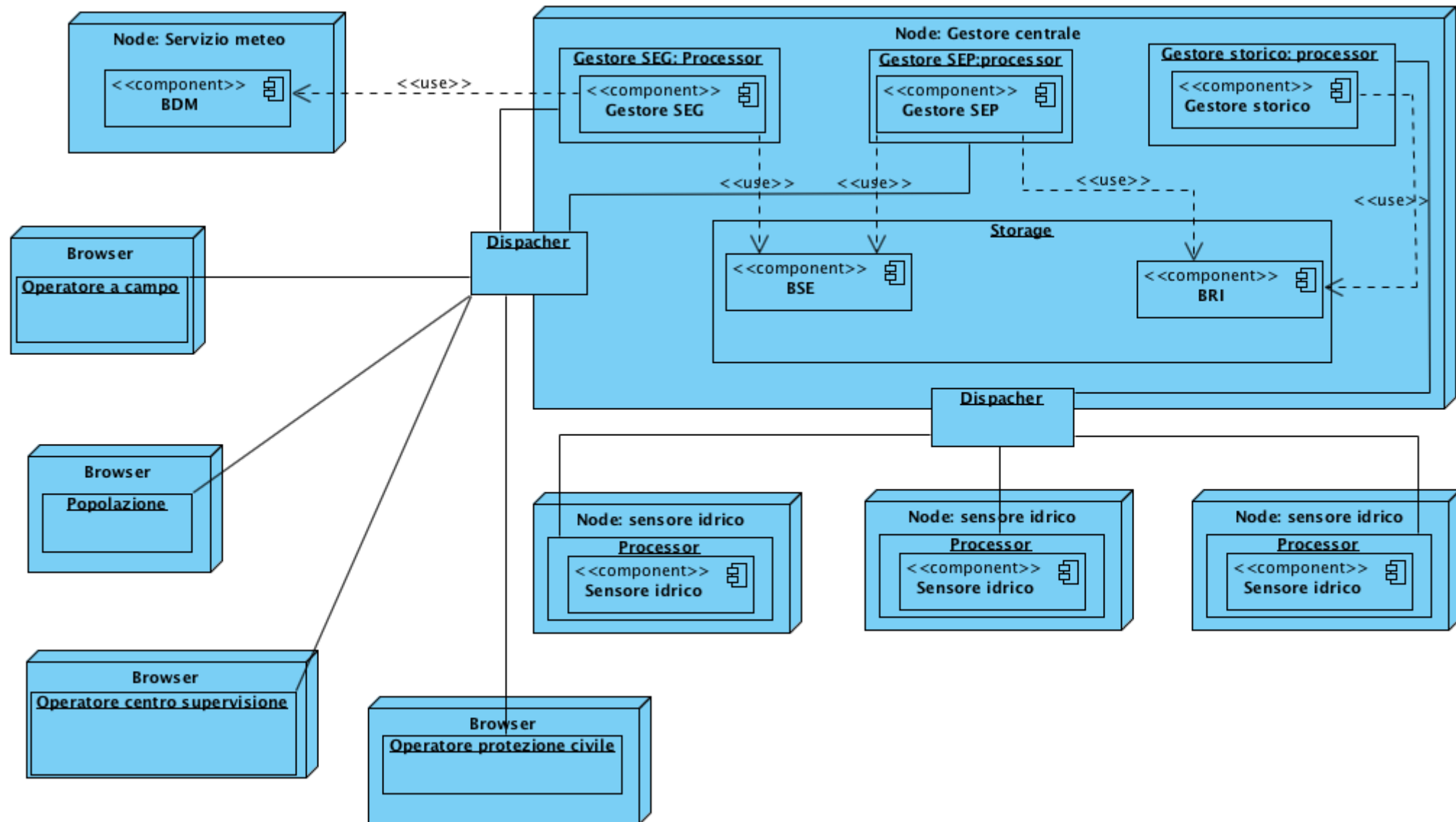
# Sequence diagrams – visualizza dati dettagliati



# Deployment architecture – basata sulla soluzione 1 dei componenti logici – non utilizzata



# Deployment architecture – basata sulla soluzione 2 dei componenti logici – non utilizzata



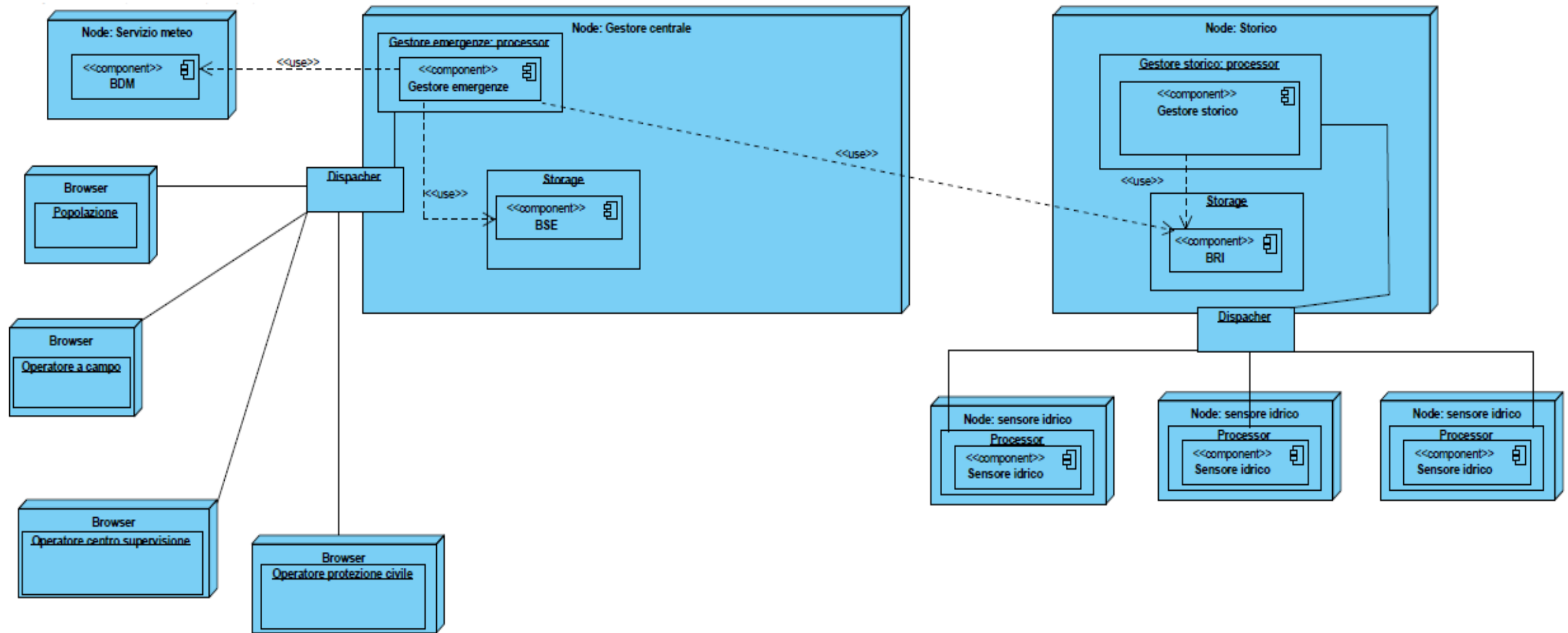
# Deployment architecture

Analizzando l'architettura logica utilizzata (soluzione 3), abbiamo realizzato due possibili architetture di deployment:

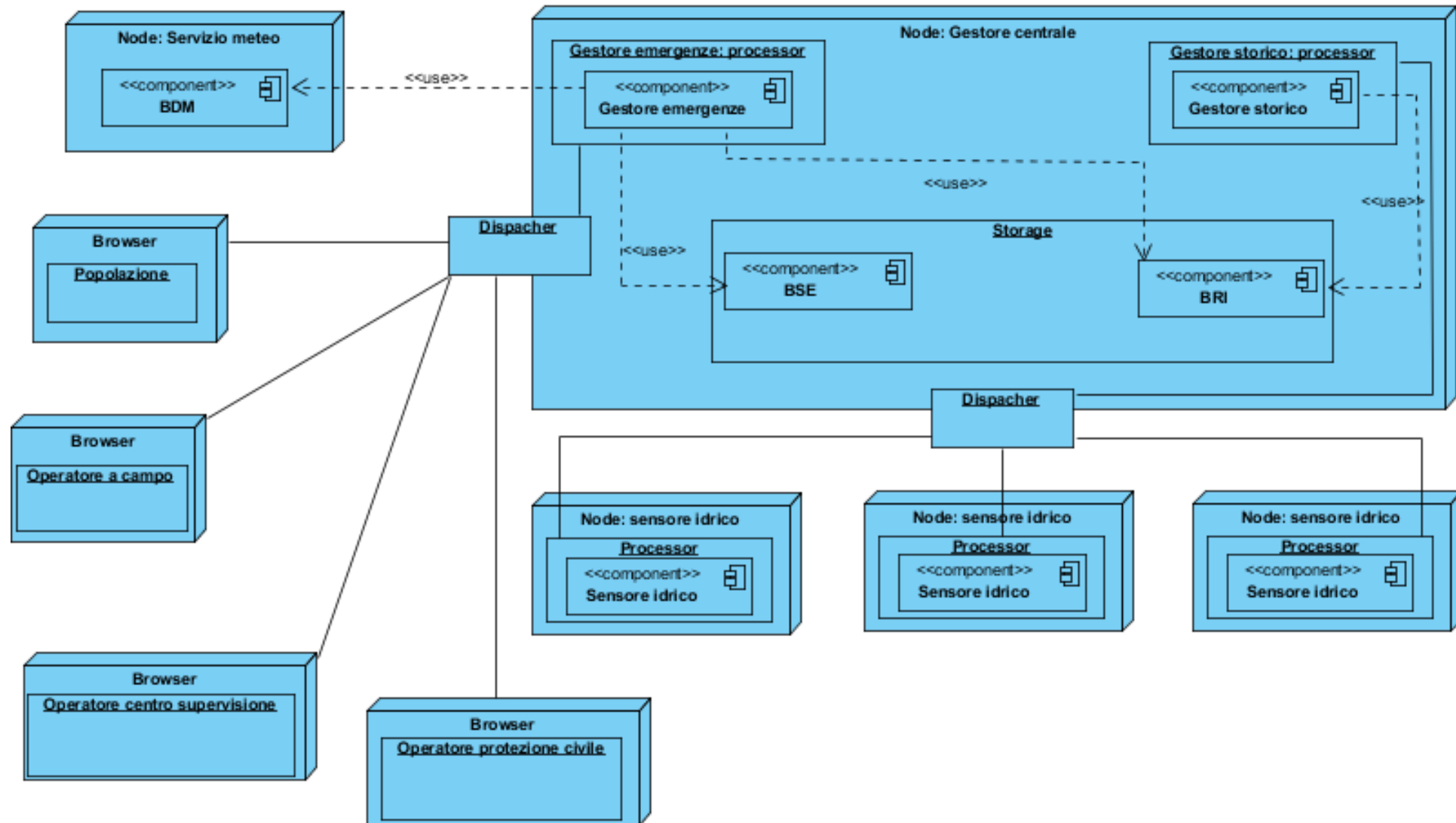
- **Gestione unico nodo**
- Gestione con due nodi (nodo centrale + nodo storico)



# Deployment architecture – basata sulla soluzione 3 dei componenti logici - non utilizzata



# Deployment architecture – basata sulla soluzione 3 dei componenti logici –utilizzata



# Architettura di deployment

## Strumenti



Sensore per la rilevazione di DI utilizzato:

### **GaugerGSM/GPRS**

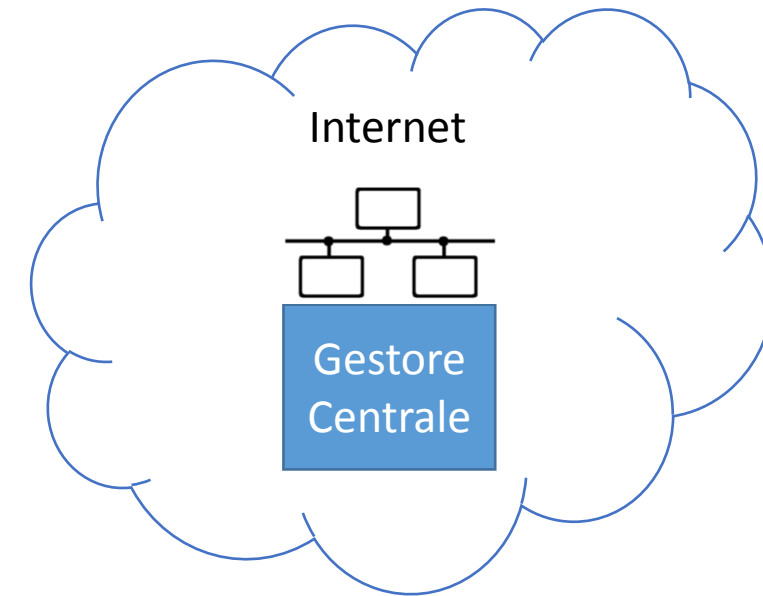
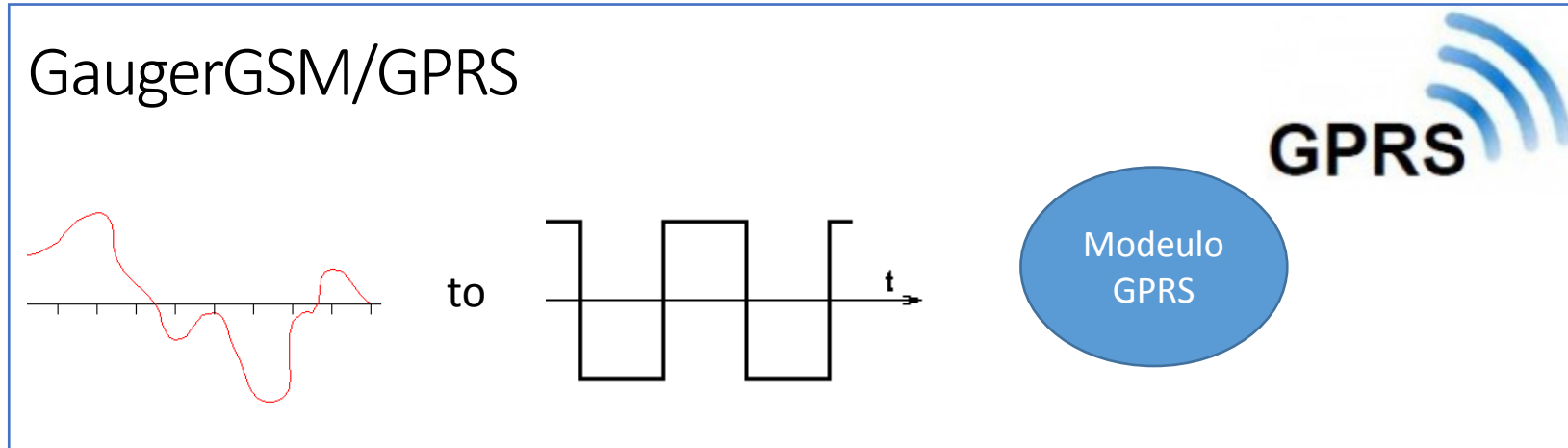
Measuring range:	8-9.5m
Power Supply (DC):	8 – 33 DC
Measurement accuracy:	0.3% / 1.5mm
Temperature:	-30° to +70°
Output:	GPRS/GSM
Price:	1400€/cad (preventivo <i>Ital Control Meters SRL</i> - febbraio 2015)

data sheet: <http://www.solidat.com/objects/DS/DS-GaugerGSM.pdf>

# Architettura di deployment

## Strumenti – funzionamento sensore

Il sensore *GaugerGSM/GPRS* rileva i dati in modo analogico e dove aver eseguito il campionamento digitale, invia i dati al server tramite *GPRS*.



# Architettura di deployment

## Strumenti

- Macchina 4 core, 7GB Ram, 320GB HDD  
Prezzo = 97,52€ / mese

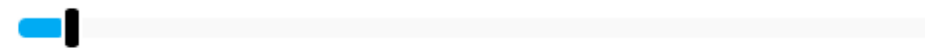


### Macchine virtuali Linux ?

Basic

Standard

Serie A



1

**€97,52**  
€0,132/ora



4 core, 7 GB di RAM

# Architettura di deployment

## Strumenti

- Macchina 2 core, 7,5GB Ram, 300GB HDD, 32SSD  
Prezzo = 101 € / mese




# Architettura di deployment

## Stima costi

Per lo sviluppo del sistema supponiamo i seguenti costi:

Analisi requisiti e creazione documenti di implementazione/architettura:

3  per 7-10 gg a 150€/h


Sviluppo del sistema partendo dalla documentazione prodotta nella fase di analisi:

5  per 90 gg a 80€/h

# Architettura di deployment

## Stima costi

Installazione di un singolo sensore:

2  per 4 ore a 30€/h

Costo per licenze *MySQL* e *Java SE Server (GlassFish)* : circa 5000€ / anno



# Architettura di deployment

## Stima costi

Riepilogo costi:

2.000 Sensori per 1400€ ** / cad =	2.800.000 €
Installazione sensori 30 € x 4h x 2p x 2000 =	480.000 €
Analisi e architettura 150 € x 8h x 3p x 10 =	36.000 €
Sviluppo e test 80 € x 8h x 5p x 90 gg=	288.000 €
Componenti HW (Azure solution) =	1170.24 € / anno
Componenti SW	5000 € / anno
Costo traffico dati sensori	6000 € circa / anno

---

**3.616.170,24 €**

\*\* sconto relativo alla quantità acquistata non incluso

NB: i costi rappresentano una stima indicativa

# Architettura dei Dati

# Schema logico

BRI

CorsiAcqua (id, denominazione)

TrattiAcqua (id, portata, *idCorsoAcqua*, *idNodoInizio*, *idNodoFine*)

NodiAcqua (id, latitudine, longitudine, regione)

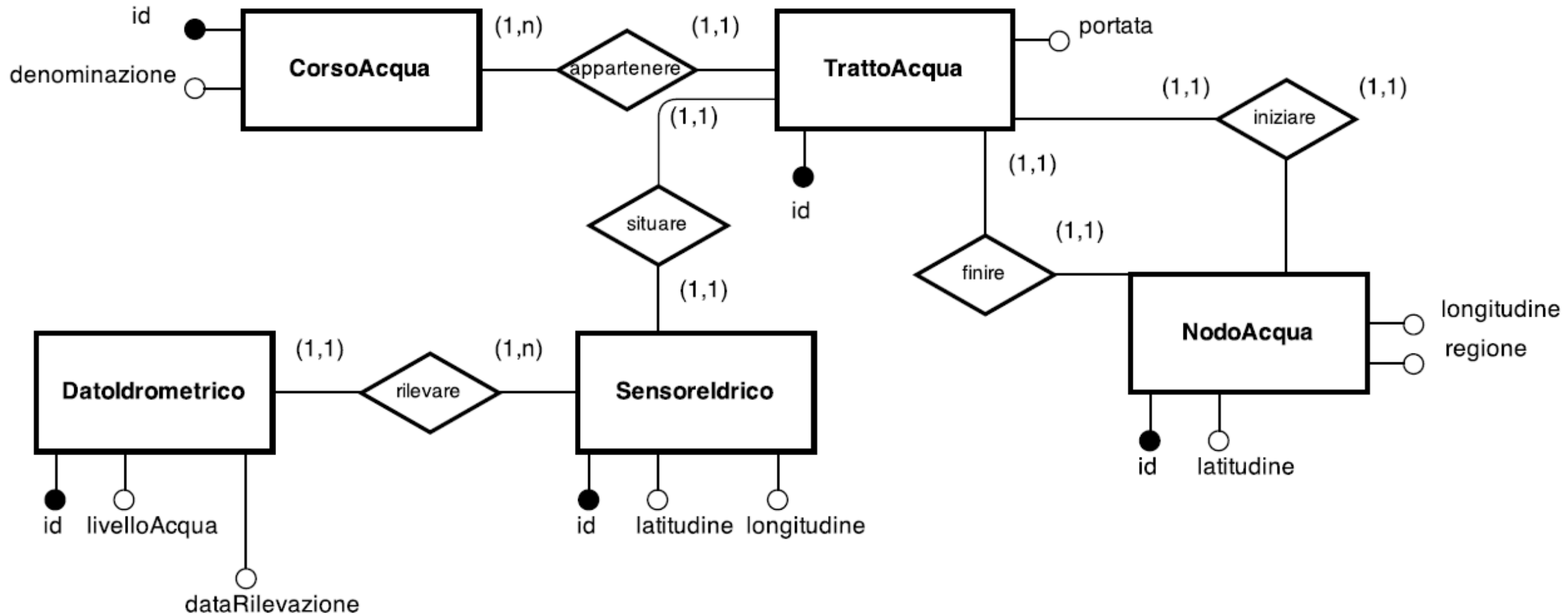
Datildrometrici (id, livelloAcqua, dataRilevazione, *idSensoreIdrico*)

Sensorildrici (id, latitudine, longitudine, *idTrattoAcqua*)

N.B. : gli attributi in *italico* sono chiavi esterne

# Schema concettuale

BRI



# Schema logico

BDM

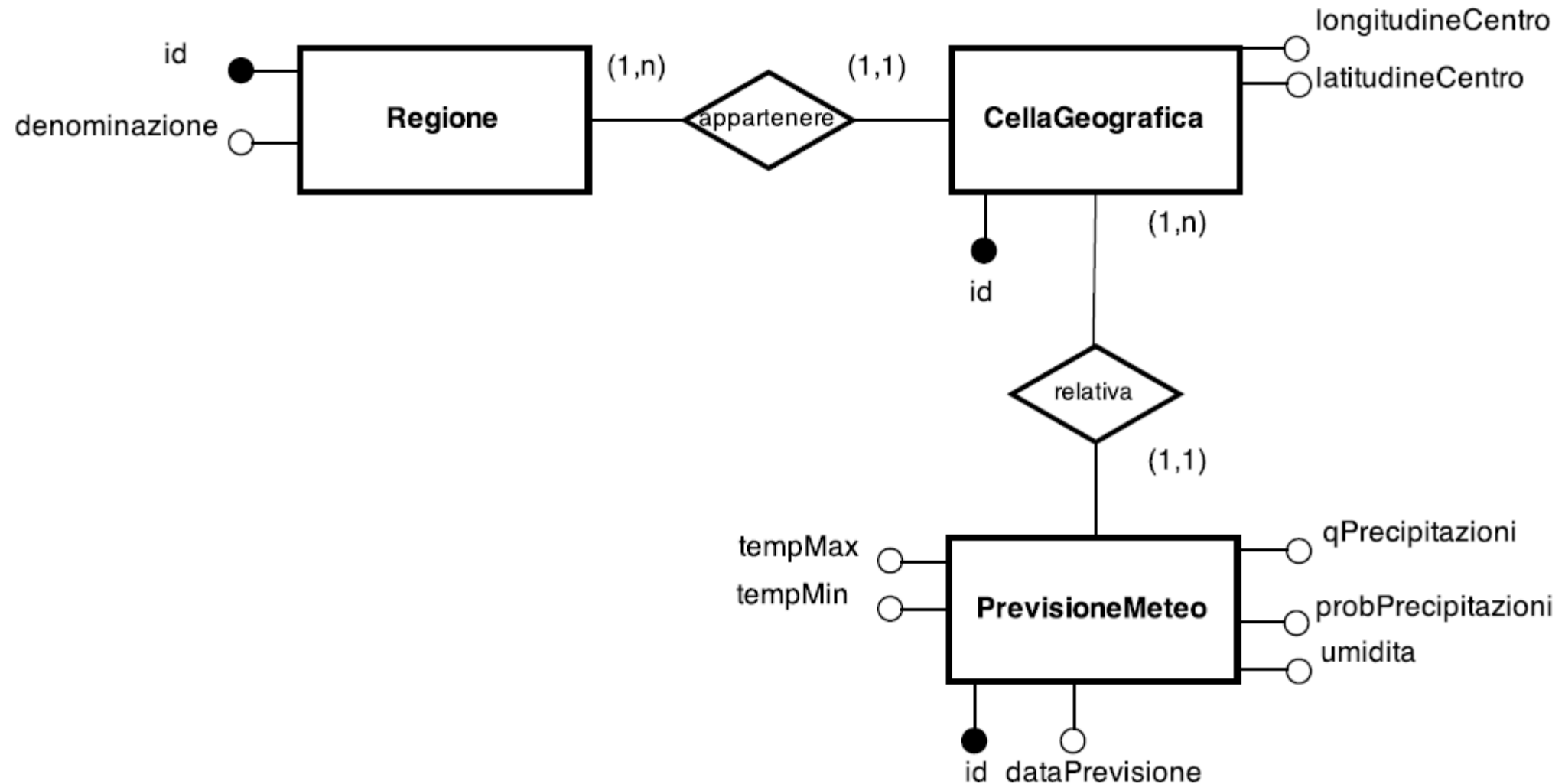
**Regioni** (id, denominazione)

**CelleGeografiche** (id, latitudineCentro, longitudineCentro, *idRegione*)

**PrevisioniMeteo** (id, dataPrevisione, umidita, probPrecipitazioni,  
qPrecipitazioni, tempMax, tempMin, *idCellaGeografica*)

# Schema concettuale

BDM



# Schema logico

## BSE

Sep (id, dataIdentificazione, dettagli)

PianificazioneSpostamenti (id, dataPianificazione, *matricolaOperatore*)

OperatoreCentroSupervisione (matricola, nome, cognome)

Previsioni (id, probPioggia, quantitaPioggia, data)

PrevisioniSensoriSep (idSep, idSensoreIdrico, idPrevisioni, dataRilevazione, di)

Sensori (id, latitudine, longitudine)

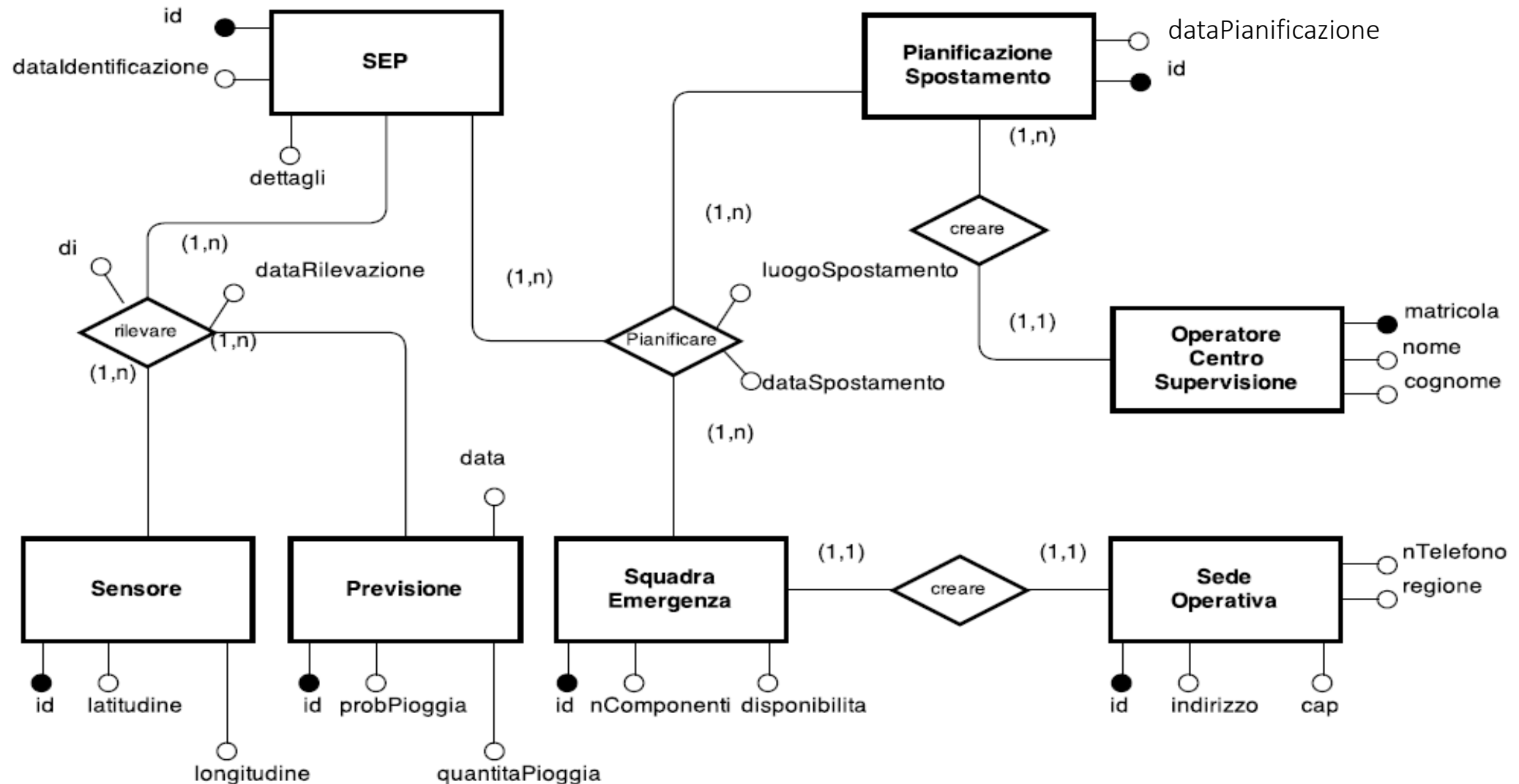
SepPianificazioniSquadra (idSep, idPianificazione, idSquadraEmergenza,  
dataSpostamento, luogoSpostamento)

SquadreEmergenza (idSquadreEmergenza, nComponenti, disponibilita, *idSedeOperativa*)

SediOperative (id, indirizzo, cap, nTelefono, regione)

# Schema concettuale

BSE





# Schema logico

## Eterogeneità e corrispondenze interschema

E/C	Schema 1	Oggetto schema 1	Schema 2	Oggetto schema 2	Tipo di E/C	Sottotipo E/C
E	BRI	Datildrometrici.livelloAcqua	BSE	PrevisioniSensoriSep.di	Schema level	Sinonimia
E	BRI	NodoAcqua.regione	BDM	Regione.denominazione	Schema level	Sinonimia
E	BRI	CorsoAcqua.denominazione	BDM	Regione.denominazione	Schema level	Omonimia
E	BRI	NodoAcqua.latitudine	BDM	CellaGeografica.latitudineCentro	Schema level	Sinonimia
E	BRI	NodoAcqua.longitudine	BDM	CellaGeografica.longitudineCentro	Schema level	Sinonimia
C	BSE	Previsioni	BDM	PrevisioniMeteo	Is - a	Iperonimia
C	BSE	Sensori	BRI	Sensorildrici	Is - a	Iperonimia

# Schema logico

## Globale

CorsiAcqua (id, denominazione)

TrattiAcqua (id, portata, *idCorsoAcqua*, *idNodoInizio*, *idNodoFine*)

NodiAcqua (id, latitudine, longitudine, *idRegione*)

Regioni (id, denominazione)

Datildrometrici (id, livelloAcqua, dataRilevazione, *idSensoreIdrico*)

SensoriIdrici (id, latitudine, longitudine, *idTrattoAcqua*)

CelleGeografiche (id, latitudineCentro, longitudineCentro, *idRegione*)

DatiSensoriPrevisioniSep (*idDatoldrometrico*, *idSep*, *idSensoreIdrico*, *idPrevisione*)

PrevisioniMeteo (id, dataPrevisione, umidita, probbPrecipitazioni, qPrecipitazioni, tempMax, tempMin, *idCellaGeografica*)

SediOperative (id, indirizzo, cap, nTelefono, *idRegione*)

Sep (id, dataIdentificazione, dettagli)

SepPianificazioniSquadra (*idSep*, *idPianificazione*, *idSquadraEmergenza*, dataSpostamento, luogoSpostamento)

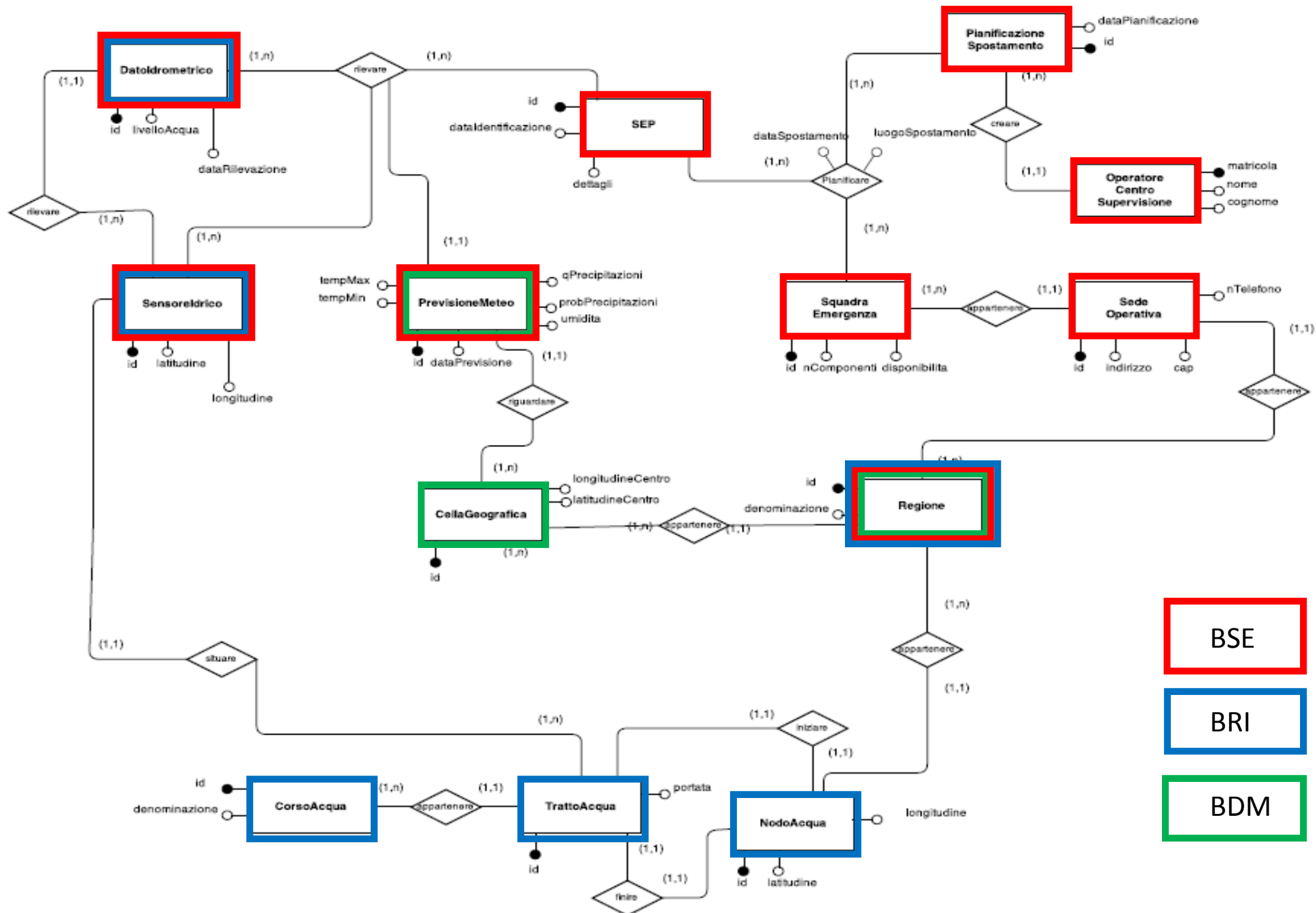
SquadreEmergenza (id, nComponenti, disponibilita, *idSedeOperativa*)

OperatoreCentroSupervisione (matricola, nome, cognome)

PianificazioneSpostamenti (id, dataPianificazione, *matricolaOperatore*)

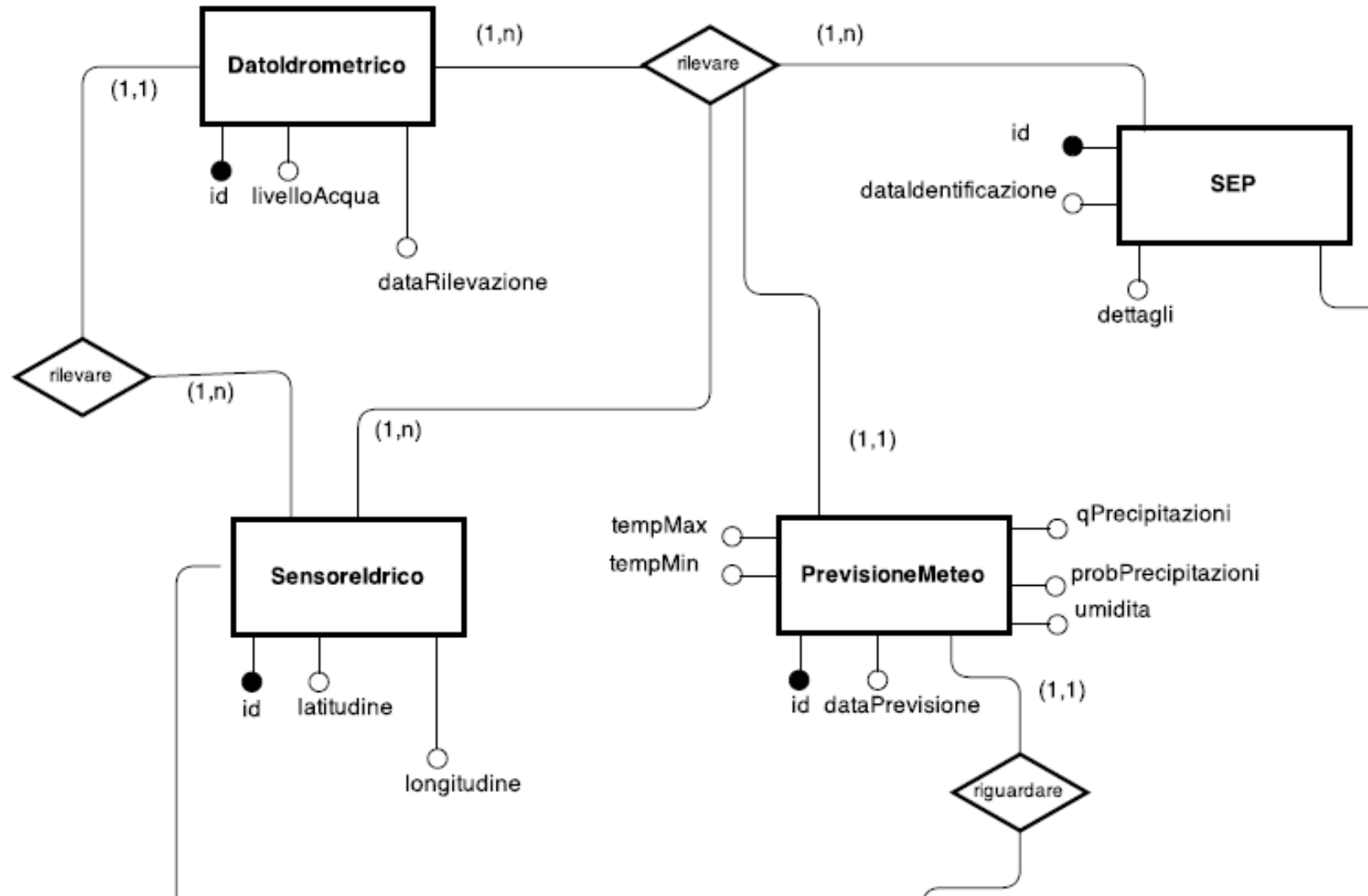
# Schema concettuale

Globale



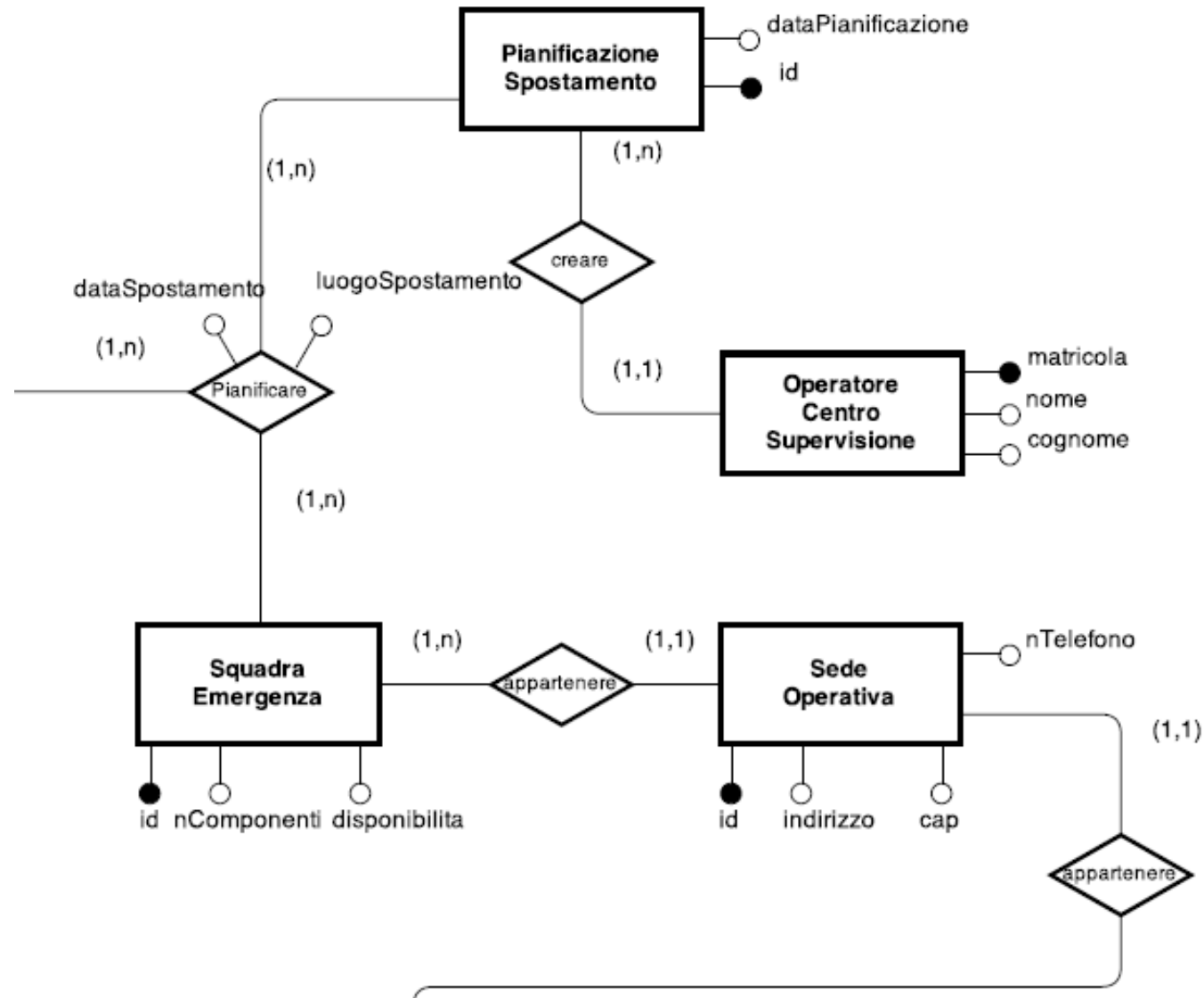
# Schema concettuale

## Globale



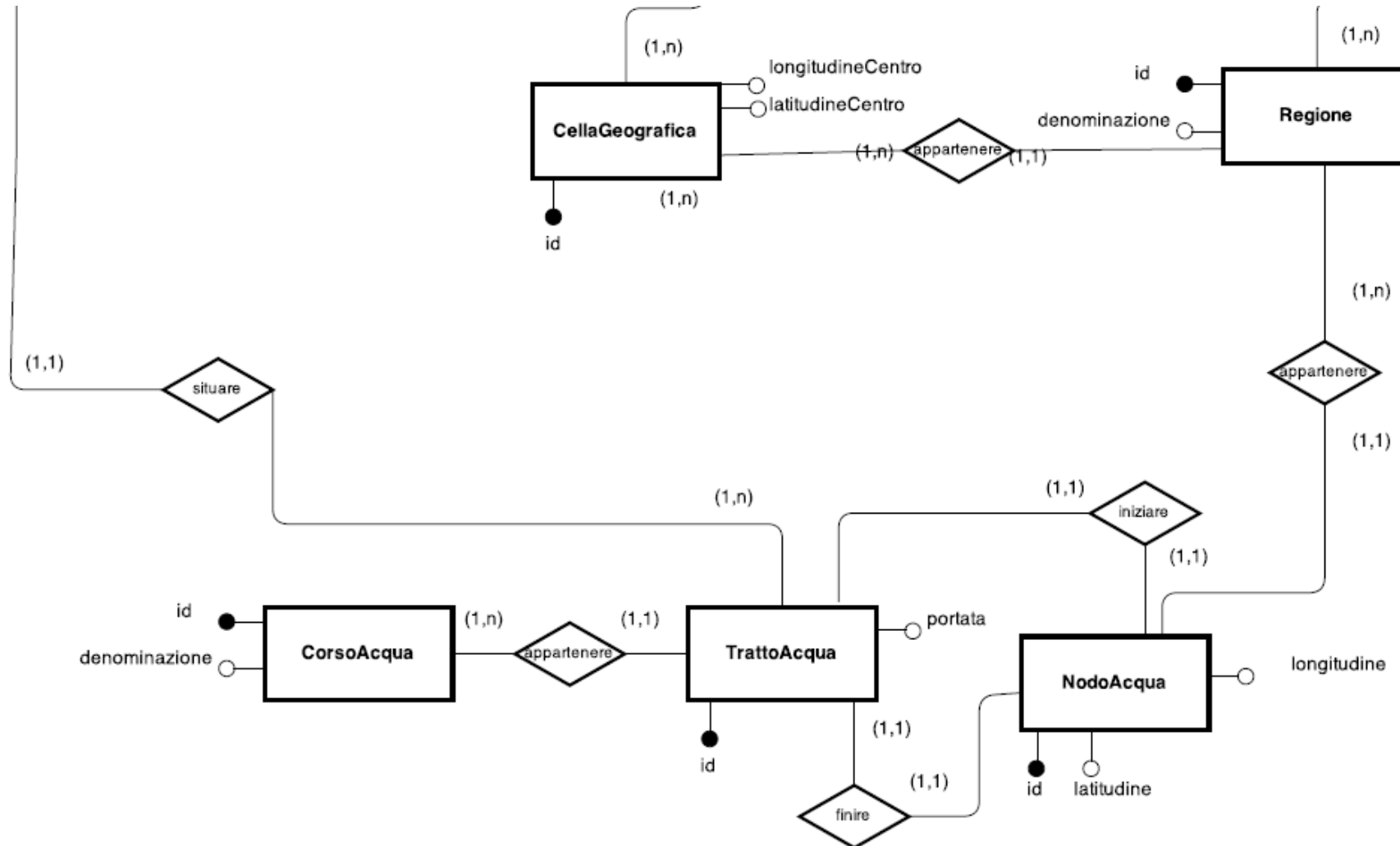
# Schema concettuale

## Globale



# Schema concettuale

## Globale



# Virtual Data Integration vs Data Warehousing

Data integration involves combining data residing in different sources and providing users with a unified view of these data.<sup>1</sup>

La modalità *Virtual data integration* comporta la creazione di uno *schema logico globale virtuale* che integra più basi di dati reali.

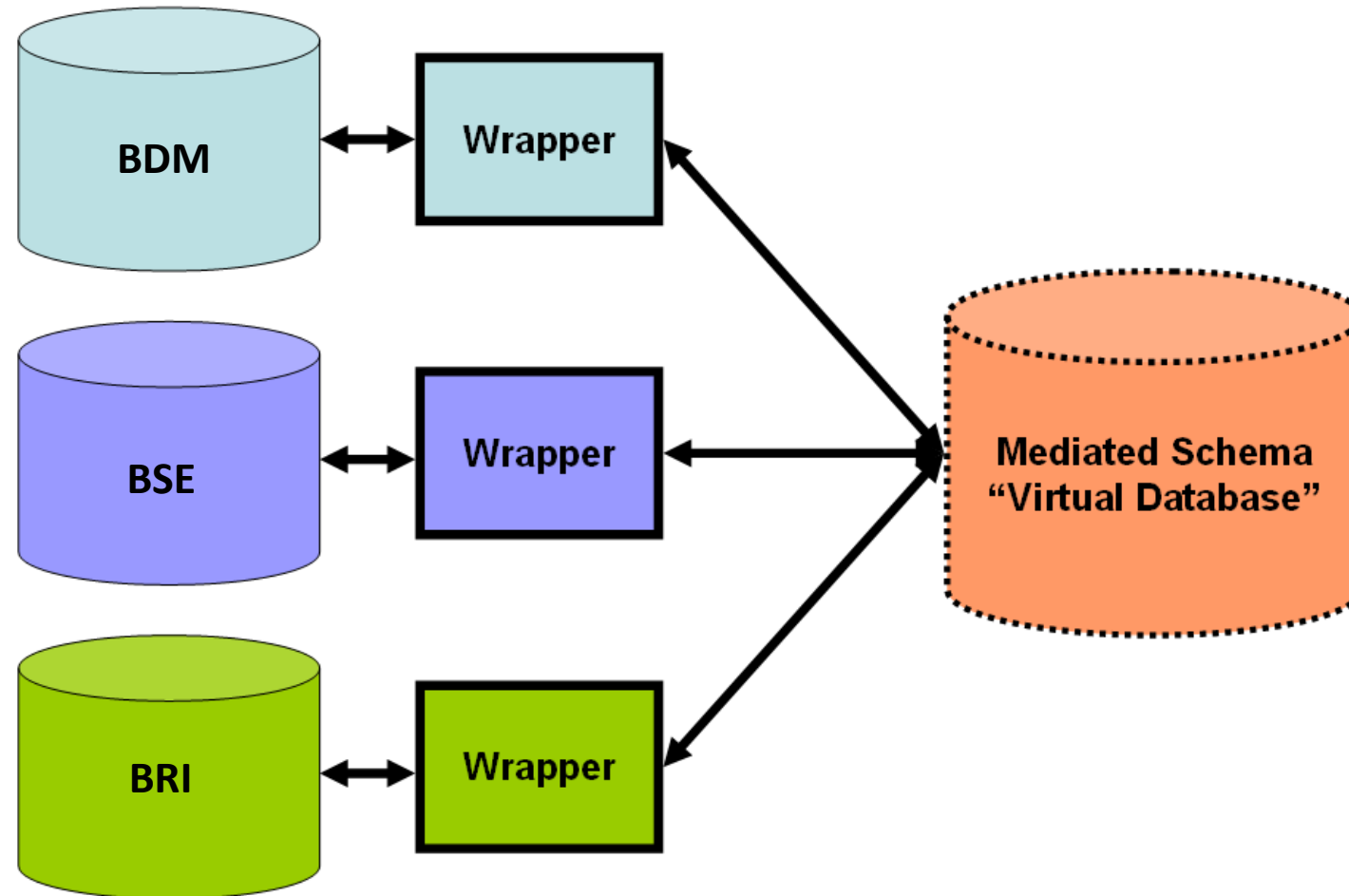
Questo non comporta una copia fisica dei dati ma solamente l'integrazione (tramite wrappers e mediatore) di dati provenienti da fonti diverse.

1. Maurizio Lenzerini (2002). "Data Integration: A Theoretical Perspective". PODS 2002. pp. 233–246.



# Virtual Data Integration vs Data Warehousing

L'esistenza di più basi di dati è trasparente all'utilizzatore dello schema virtuale.



# Virtual Data Integration vs Data Warehousing

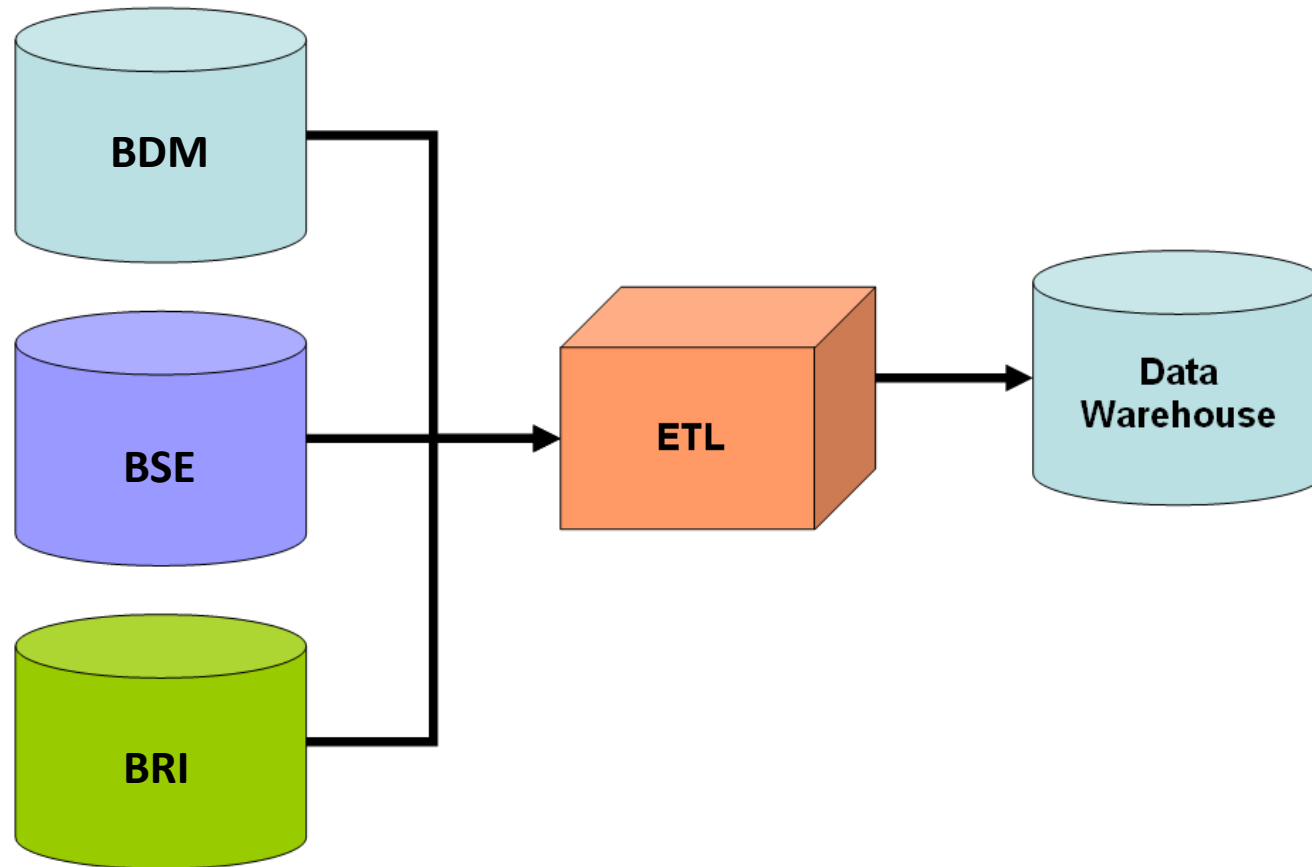
Si può utilizzare, per data warehousing, la modalità ETL – *Extract, Transform, Load*, che comprende il processo di estrazione, trasformazione e caricamento di dati su di un sistema di sintesi.

Dopo il processo di estrazione (cioè il recupero di dati da un sistema esterno), i dati vengono trasformati, ed esempio:

- Normalizzazione dei dati
- Selezione dei dati
- Computazione di nuovi dati
- Raggruppamento
- Ecc...

# Virtual Data Integration vs Data Warehousing

Dopo la trasformazione, i dati vengono caricati (load) sulle tabelle del nuovo sistema di sintesi (comporta la copia fisica sul nuovo sistema)



# Virtual Data Integration vs Data Warehousing

L'integrazione fra le basi dai BRI, BSE e BDM può essere implementata via VDI o data warehouse.

Integrazione via **VDI**:

**Pro:**

Le tuple presenti nelle basi dati non vengono replicate *fisicamente* su di una nuova base dati.

**Contro:**

Ogni interrogazione comporta interrogazioni verso  $n$  basi di dati.

Nel nostro sistema essendo BDM esterna, non è sempre possibile conoscerne lo stato e le tempistiche di interrogazione.

# Virtual Data Integration vs Data Warehousing

L'integrazione fra le basi dei BIR, BSE e BDM può essere implementata via VDI o data warehouse.

Integrazione **data warehouse** (via ETL):

**Pro:**

Le interrogazioni alle tuple sono sempre possibili indipendentemente dallo stato dei data source.

È possibile trasformare i dati. (esempio normalizzare le misurazioni dei sensori)

**Contro:**

È necessario creare una nuova base di dati e gestire lo scheduling delle operazioni di extract, transform e load.

# Mapping

## CorsiAcqua

```
CREATE VIEW CorsiAcqua AS  
SELECT *  
FROM BRI.CorsiAcqua
```

## TrattiAcqua

```
CREATE VIEW TrattiAcqua AS  
SELECT *  
FROM BRI.TrattiAcqua
```

# Mapping

## NodiAcqua

```
CREATE VIEW NodiAcqua AS
SELECT BRI.NodoAcqua.id, BRI.NodoAcqua.latitudine, BRI.NodoAcqua.longitudine,
       BDM.Regioni.id
FROM BRI.NodoAcqua, BDM.Regioni
WHERE BRI.NodoAcqua.regione = BDM.Regioni.denominazione
```

## Regioni

```
CREATE VIEW Regioni AS
SELECT *
FROM BDM.Regioni
```

# Mapping

## Datildrometrici

```
CREATE VIEW Datildrometrici AS  
SELECT *  
FROM BRI.Datildrometrici
```

## Sensorildrici

```
CREATE VIEW Sensorildrici AS  
SELECT *  
FROM BRI.Sensorildrici
```



# Mapping

## CelleGeografiche

```
CREATE VIEW CelleGeografiche AS  
SELECT *  
FROM BDM.CelleGeografiche
```

## DatiSensoriPrevisioniSep

```
CREATE VIEW DatiSensoriPrevisioniSep AS  
SELECT BRI.Datildrometrici.id, BSE.PrevisioniSensoriSep.idSep,  
       BSE.PrevisioniSensoriSEP.idSensoreIdrico, BSE.PrevisioniSensoriSEP.idPrevisioni  
FROM BRI.Datildrometrici, BSE.PrevisioniSensoriSep  
WHERE BSE.PrevisioniSensoriSep.di = BRI.Datildrometrici.livelloAcqua  
AND BSE.PrevisioniSensoriSep.dataRilevazione = BRI.Datildrometrici.data
```

# Mapping

## PrevisioniMeteo

```
CREATE VIEW PrevisioniMeteo AS  
SELECT *  
FROM BDM.PrevisioniMeteo
```

## SediOperative

```
CREATE VIEW SediOperative AS  
SELECT BSE.SediOperative.id, BSE.SediOperative.indirizzo, BSE.SediOperative.cap,  
       BSE.SediOperative.nTelefono, BDM.Regione.id  
FROM BSE.SediOperative, BDM.Regione  
WHERE BSE.SediOperative.regione = BDM.Regione.denominazione
```

# Mapping

## Sep

```
CREATE VIEW Sep AS  
SELECT *  
FROM BSE.Sep
```

## SepPianificazioniSpostamenti

```
CREATE VIEW SepPianificazioniSpostamenti AS  
SELECT *  
FROM BSE.SepPianificazioniSpostamenti
```

# Mapping

## SquadreEmergenza

```
CREATE VIEW SquadreEmergenza AS  
SELECT *  
FROM BSE.SquadreEmergenza
```

## OperatoreCentroSupervisione

```
CREATE VIEW OperatoreCentroSupervisione AS  
SELECT *  
FROM BSE.OperatoreCentroSupervisione
```

# Mapping

## PianificazioniSpostamento

```
CREATE VIEW PianificazioniSpostamento AS  
SELECT *  
FROM BSE.PianificazioniSpostamento
```

# Query schema globale

«Data la denominazione di un fiume ed un intervallo di date (data inizio e data fine), estrarre le previsioni dettagliate per ogni SEP verificatasi per il fiume richiesto nell'intervallo temporale dato».

Parametri input :

- Denominazione corso d'acqua: @nomefiume
- Data inizio: @datainizio
- Data fine: @datafine

# Query schema globale

```
SELECT Sep.data, Sep.dettagli, Previsione.*  
FROM CorsoAcqua, TrattiAcqua, Sensorildrici, DatiSensoriPrevisioniSep, Sep,  
     Previsioni  
WHERE CorsoAcqua.id = TrattiAcqua.idCorsoAcqua  
AND TrattiAcqua.id = Sensorildrici.idTrattoAcqua  
AND Sensorildrici.id = DatiSensoriPrevisioniSep.idSensoreIdrico  
AND DatiSensoriPrevisioniSep.idSep = Sep.id  
AND DatiSensoriPrevisioniSep.idPrevisione = Previsioni.id  
WHERE CorsoAcqua.denominazione = @nomefiume  
AND Sep.dataIdentificazione BETWEEN @datainizio AND @datafine
```

# Query schema globale - unfolding

```
SELECT BSE.Sep.data, BSE.Sep.dettagli, BDM.PrevisioniMeteo.*
FROM BRI.CorsoAcqua, BRI.TrattiAcqua, BRI.Datildrometrici, BSE.Sensorildrici,
      BSE.SensoriPrevisioniSep, BSE.Sep, BDM.PrevisioniMeteo
WHERE BRI.CorsoAcqua.id = BRI.TrattiAcqua.idCorsoAcqua
AND BRI.TrattiAcqua.id = BRI.Sensorildrici.idTrattoAcqua
AND BRI.Sensorildrici.id = BSE.SensoriPrevisioniSep.idSensoreldrico
AND BSE.SensoriPrevisioniSep.idSep = BSE.Sep.id
AND BSE.SensoriPrevisioniSep.idPrevisione = BDM.Previsioni.id
AND BSE.PrevisioniSensoriSep.di = BRI.Datildrometrici.livelloAcqua
AND BSE.PrevisioniSensoriSep.dataRilevazione = BRI.Datildrometrici.data
AND BRI.CorsoAcqua.denominazione = @nomefiume
AND BSE.Sep.dataIdentificazione BETWEEN @datainizio AND @datafine
```



# Open Data

I valori dei dati nello storico e i relativi corsi d'acqua presenti nella base dati BRI saranno resi pubblici e fruibili via web browser.

I dati pubblicati rispettano i vincoli di privacy e fruibilità degli open data e sono esposti in formato JSON via REST API.

```
{  
  "IDSensore":1,  
  "CorsoAcqua":"Adda",  
  "DataMisurazione":"2015-02-19T14:35Z",  
  "Valore":56.78,  
  "Latitudine":45.4654219,  
  "Longitudine":9.1859243  
}
```

# Open Data

I dati sui sensori esposti rispettano i principi degli Open Data:

- Completati: sono completi di tutte le informazioni per l'utilizzo anche offline
- Primari: hanno granularità tale che ne permette l'integrazione con altre applicazioni
- Tempestivi: rappresentazione real time dello storico
- Accessibili: disponibili via REST API
- Non proprietari: i dati sono processabili da applicativi open source
- Non discriminatori: non sono previsti meccanismi di registrazione per l'utilizzo dei dati (es: API KEY)

# Open Data

I dati sono pubblicati sotto licenza *Italian Open Data Licenses 2.0* che ne permette l'utilizzo, ma obbliga l'utilizzatore a citare il Licenziante.

Licenza: <http://www.dati.gov.it/iodl/2.0/>

I dati pubblicati possono essere utilizzati ad esempio dal corso di laurea di statistica o di geologia (analisi ed inferenza statistica sui dati) oppure da agenzie che si occupano di gestione territoriale.

# Considerazioni

# Considerazioni – Architettura dati

L'architettura dati implementata è centralizzata, una possibile alternativa è quella di distribuire le basi dati BRI e BSE.

La scelta di una base dati distribuita introduce problemi di *replicazione dati* e *distribuzione dei frammenti*.

La scelta di un architettura centralizzata riduce i costi della gestione dei dati (replicazione/frammentazione, mutua esclusione).

NB: per costi si intendono le risorse utilizzare per svolgere un operazione

# Frammentazione BSE

Le informazioni che è possibile *frammentare orizzontalmente*, sono le seguenti :

- Squadre di emergenza
- Pianificazione spostamento
- Sede operativa
- Operatore centro di supervisione
- Sensore idrico
- SEP

Supponiamo una *frammentazione orizzontale* su 20 regioni. (quindi la base dati sarà distribuita su 20 nodi)

# Replicazione BSE

Le informazioni per cui è necessaria la *replicazione*, sono le seguenti :

- SEP

Le *situazioni di emergenza potenziale* comuni a più regioni, sono informazioni da replicare su ogni nodo dell'istanza distribuita.

La gestione della replicazione delle informazioni, implica l'utilizzo di una strategia di **mutua esclusione** per garantire la **consistenza** dei dati.

NB: la pianificazione di spostamenti per gestire una SEP che comprende più regioni, è da considerarsi una situazione straordinaria

# Frammentazione BRI

Le informazioni che è possibile *frammentare orizzontalmente*, sono le seguenti :

- Sensore idrico
- Dato idrometrico
- Nodo acqua
- Tratto acqua
- Corso acqua

Supponiamo una *frammentazione orizzontale* su 20 regioni. (quindi la base dati sarà distribuita su 20 nodi)

NB: assumiamo non ci siano sensori e nodi sul confine regionale



# Replicazione BRI

Le informazioni per cui è necessaria la *replicazione*, sono le seguenti :

- Tratto acqua
- Corso acqua

I corsi d'acqua con i relativi tratti, comuni a più regioni, devono essere replicati fra i *nodi DB* delle regioni coinvolte.

Esempio: il corso d'acqua *Po* è replicato nelle basi dati delle regioni Piemonte, Lombardia, Emilia Romagna e Veneto. (il tratto fra Piacenza e Cremona è condiviso fra Lombardia ed Emilia Romagna)

# Considerazioni – Architettura software

Le possibili architetture software analizzate sono le seguenti:

- Unico nodo di *Gestione Centrale* per l'intero sistema
- Un nodo di *Gestione Centrale* per ogni regione

## Pro:

La soluzione ottimale per il nostro sistema è l'utilizzo di un unico nodo di Gestione Centrale in quanto il numero di dati da monitorare (sensori) è relativamente piccolo (nella nostra stima 2000 sensori) quindi la duplicazione di hw e sw comporterebbe costi (in termini di denaro, sviluppo e manutenzione) maggiori. (non giustificabili da un incremento delle prestazioni del sistema)

# Considerazioni – Architettura software

## Contro:

L'utilizzo di un solo nodo centrale comporta una difficoltà maggiore nel controllo dello stato del sistema, in quanto il malfunzionamento di un componente, se non gestito nel modo corretto (con procedure di recovery e fault tolerance), comprometterebbe la stabilità dell'intero sistema.

L'utilizzo di nodi computazionali distribuiti non preclude il funzionamento dell'intero sistema in seguito ad un malfunzionamento localizzato.

# Considerazioni – Architettura software

Le possibili configurazioni del gestore centrale sono le seguenti:

- Gestore centrale suddiviso nelle componenti *Gestore Storico* e *Gestore Emergenze*
- Gestore centrale formato da *un solo* componente
- Gestore centrale suddiviso nelle componenti *Gestore SEP*, *Gestore SEG* e *Gestore Storico*

Analizzando le tre diverse configurazioni, la prima è da considerarsi migliore.  
(vedi slide sui *footprint*)

# Considerazioni – Architettura dati

La scelta di una base dati centralizzata è dovuta alla presenza di numerose letture verso lo storico delle rilevazioni idrometriche.

L'algoritmo viene eseguito una volta all'ora; la presenza nel sistema di una base dati distribuita comporterebbe una lettura massiva da ogni nodo dell'istanza.

(considerando che la soluzione sw adottata è centralizzata, devo recuperare lo storico per 2000 sensori distribuiti in 20 nodi!!!)

# Considerazioni – Varie

La trasmissione dei dati tra *sensori* e *gestore centrale* avviene tramite segnale GPRS. (*wireless*)

La scelta di questo tipo di trasmissione permette il posizionamento dei sensori anche in zone difficilmente raggiungibili da rete internet cablata.

Un altro tipo di approccio è quello di utilizzare la trasmissione via cavo con conseguente riduzione dei costi (un modulo *cable internet* è meno costoso di un *modulo GPRS*) ma si ottengono limitazioni sul posizionamento dei sensori.

NB: assumiamo una copertura completa del segnale GPRS

# Evoluzioni future

Il sistema può essere migliorato sviluppando una o più delle seguenti features:

- Gestione automatica della pianificazione degli spostamenti in caso di SEP
- Introduzione di nuove tipologie di sensori per una maggiore probabilità di identificazione SEP (esempio sensore sismico, geotermico ecc)
- Introduzioni di nuovi metodi di segnalazione SEG (non solo da operatori a campo)

End