POLITECNICO
DI MILANO

CISIS
Centro Interregionale
per i Sistemi informatici, geografici e statistici
Comitato permanente
per i Sistemi Informativi Geografici

Implementazione delle proprietà geometriche del GeoUML

1 febbraio 2012

SpatialDBgroup

SpatialDBgroup@polimi.it

http://SpatialDBgroup.polimi.it

Autori	Politecnico di Milano – Spatial DB Group	Giuseppe Pelagatti (coordinatore), Alberto Belussi, Jody Marca, Mauro Negri
Coordinamento delle attività	CISIS – CPSG Comitato di Progetto	Maurizio De Gennaro (Regione del Veneto – coordinatore tecnico), Massimo Attias (CISIS-referente area geografica e progetti) Stefano Olivucci (Regione Emilia-Romagna), Raffaella Gelleti, Marco Lunardis, Massimo Zia (Regione Friuli Venezia Giulia), Massimiliano Basso, Alessandra Chiarandini (INSIEL-Regione FVG), Simone Patella (Regione Lazio), Gianbartolomeo Siletto (Regione Piemonte), Mauro Vasone (CSI Piemonte), Marco Guiducci, Andrea Peri (Regione Toscana), Gianfranco Amadio, Domenico Bertoldi, Gianluca Riscaio, Sandra Togni (Regione Umbria), David Freppaz (Regione Valle d'Aosta), Virgilio Cima, Umberto Trivelloni (Regione del Veneto), Leonardo Donnaloia, Claudio Mazzi, Pierpaolo Milan (CISIS)
	CISIS –CPSG Struttura di supporto interna	Massimo Attias, (coordinatore struttura), Leonardo Donnaloia, Claudio Mazzi, Pierpaolo Milan, Antonio Rotundo (CISIS)

INDICE

1. INTRODUZIONE

Il linguaggio GeoUML definisce le proprietà geometriche e topologiche facendo riferimento allo spazio euclideo continuo, nel quale le coordinate sono numeri reali.
Le implementazioni dei Data Product che devono essere conformi a specifiche GeoUML sono basate su una rappresentazione vettoriale delle geometrie con coordinate rappresentate da numeri in precisione finita. Per questo motivo è necessario definire delle ***regole di implementazione*** che stabiliscono come una geometria vettoriale finita deve essere strutturata per rappresentare correttamente una specifica GeoUML.
Purtroppo, esistono numerosi problemi, tutti legati in ultima analisi alla rappresentazione finita dei numeri, che possono rendere la valutazione delle proprietà topologiche di un data product ambigua, cioè tale per cui diversi sistemi potrebbero valutare in maniera diversa le relazioni topologiche esistenti tra le geometrie.
Il contesto di riferimento è mostrato in figura 1.1: un sistema sorgente S produce un Data product DP che viene controllato dal GeoUML Validator V e caricato in un sistema destinazione D.
Le regole di implementazione mirano in particolare a evitare problemi nelle seguenti situazioni:

1. controllo dei dati da parte del GeoUML Validator: si deve evitare che si verifichino situazioni ambigue rispetto alla specifica;
2. lettura dei dati nel sistema destinazione: si deve evitare che i dati caricati nel sistema D abbiano proprietà topologiche diverse da quelle riscontrate dal validatore V.

Ambedue questi aspetti riguardano l'uso del GeoUML Validator: il primo in maniera ovvia, il secondo più subdolamente, perché richiede che il tipo di validazione attuata dal GeoUML Validator permetta al sistema D di operare senza problemi considerandoli topologicamente corretti. Questa seconda proprietà non potrà prescindere da alcune ipotesi minimali rispetto al comportamento del sistema D.
In altri termini, le regole di implementazione devono mirare a garantire non solo che i dati possano essere validati ma anche che la correttezza dei dati validati sia "stabile" rispetto all'uso dei dati stessi in un altro sistema.

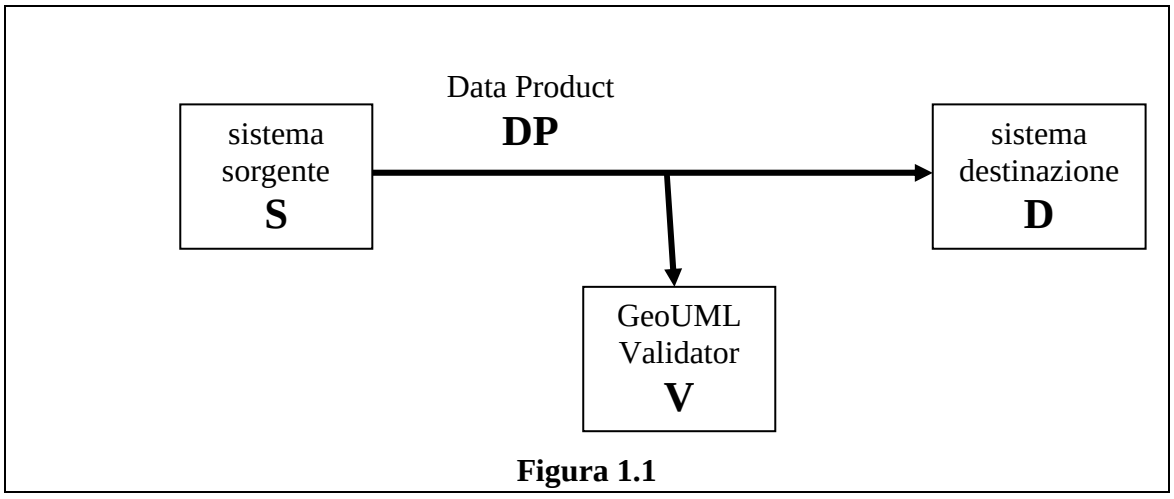


Figura 1.1

Si osservi che la configurazione rappresentata in figura 1.1 è una semplificazione rispetto a un obiettivo più generale, che consiste nel voler garantire che le proprietà topologiche verificate dal Validator siano utilizzabili da applicazioni e servizi posti a valle del sistema D, implicando quindi ulteriori trasferimenti di dati e l'impiego di

sistemi e algoritmi di vario genere; come i seguenti capitoli cercheranno di mostrare, l'ottenimento di un simile risultato è difficile e richiede comunque di porre delle regole minimali che tutti i sistemi coinvolti dovranno osservare.

I capitoli 2, 3 e 4 analizzano le possibili fonti di problemi nella valutazione delle proprietà topologiche; il capitolo 5 suggerisce alcune regole miranti ad eliminare le conseguenze di tali problemi.

2. RAPPRESENTAZIONE FINITA DELLE COORDINATE

2.1 Rappresentazioni e griglia

I numeri reali non possono essere direttamente ed esattamente rappresentati nel calcolatore; deve essere quindi utilizzata una tecnica di rappresentazione finita ed approssimata. Esistono molte diverse tecniche, che chiameremo ***Rappresentazioni*** *(finite dei numeri reali*).

Indichiamo con R_1, R_2,… R_i,.. le *rappresentazioni* adottate in diversi contesti. Ad esempio, nel nostro contesto:

- R_S è la rappresentazione utilizzata nel contesto sorgente
- R_{DP} è la rappresentazione utilizzata nel data product DP
- R_D è la rappresentazione utilizzata nel sistema destinazione dei dati
- R_V è la rappresentazione utilizzata nel GeoUML Validator

Dato un numero reale n, indichiamo con $R_i(n)$ il ***valore rappresentato*** dalla rappresentazione R_i applicata ad n.

In generale, avremo che $R_i(n) \neq n$. Questa disuguaglianza esprime il fatto che la rappresentazione finita è approssimata. Inoltre, le diverse rappresentazioni introducono approssimazioni diverse, quindi in generale può accadere che $R_i(n) \neq R_j(n)$.

Data una rappresentazione R_i, chiamiamo ***Griglia(***R_i) l'insieme dei valori rappresentabili (il termine Griglia è utilizzato perché nel nostro contesto più che i singoli valori numerici ci interessano le coppie o terne di valori che rappresentano le coordinate di uno spazio discreto bi- o tridimensionale).

2.2 Caratteristiche di una rappresentazione

Esistono molti tipi di rappresentazioni, ma per molti aspetti applicativi le differenze di dettaglio possono essere trascurate e una rappresentazione può essere caratterizzata da due aspetti:

1. la ***Base*** di una rappresentazione, che possiamo in pratica ridurre alle due alternative di rappresentazione ***binaria*** e di rappresentazione ***decimale***. Si tenga presente che le rappresentazioni utilizzate internamente dai sistemi sono quasi sempre di tipo binario, con una prevalenza al momento per la rappresentazione binaria floating point FP64, che è supportata dalla grandissima maggioranza dell'Hardware.
2. la ***Risoluzione*** di una rappresentazione, che indica la dimensione dell'intervallo tra due numeri consecutivi della griglia; nelle coordinate tale dimensione rappresenta una lunghezza e l'unità di misura è costituita dai metri. La risoluzione di una rappresentazione dipende generalmente dalla dimensione dell'intervallo di valori che deve essere rappresentato, che chiameremo ***extent***. Per semplicità in questo documento considereremo prefissato l'extent, in modo da poter parlare di risoluzione della rappresentazione.

Per esplicitare la base e la risoluzione di una rappresentazione R_i useremo la notazione seguente:

- Base(R_i) = 2 oppure 10

- Risoluzione(R_i)=k, dove k rappresenta l'esponente della più piccola frazione rappresentata; in pratica è il numero di cifre (binarie o decimali) dopo la virgola.

Esempi:

- <Base(R_i) =10, Risoluzione(R_i)=3> definisce R_i come una rappresentazione decimale con 3 cifre decimali di precisione – risoluzione 10^{-3} metri (un millimetro)
- <Base(R_i) =2, Risoluzione(R_i)=10> definisce R_i una rappresentazione binaria con 10 cifre binarie di precisione – risoluzione 2^{-10} (poco meno di un millimetro)

Le singole rappresentazioni possono variare molto nei dettagli, ma per gli aspetti che dovremo discutere potremo in generale astrarre da tali dettagli e concentrarci sulle 2 caratteristiche evidenziate.

Risoluzione delle rappresentazioni in virgola mobile:
Le rappresentazioni in virgola mobile hanno una risoluzione variabile nell'ambito dell'extent, perché la risoluzione dipende dalla grandezza della parte intera del numero. Nel seguito tratteremo le rappresentazioni in virgola mobile come se fossero rappresentazioni in virgola fissa caratterizzate dalla risoluzione peggiore nell'ambito dell'extent, cioè quella della rappresentazione del numero con parte intera più grande.

Esempio: FP64

La rappresentazione FP64, attualmente la più diffusa perché supportata da quasi tutti i tipi di Hardware, ha una struttura complessa, in virgola mobile, su un numero fisso di bit (64), quindi la sua risoluzione è variabile e dipende dalla grandezza della parte intera del numero.
Nel seguito applichiamo l'ipotesi esposta sopra, cioè consideriamo FP64 come una rappresentazione a virgola fissa con risoluzione costante e uguale a quella delle coordinate più grandi da rappresentare.
In particolare, nel seguito consideriamo come extent il Bounding Box UTM32/WGS84 dell'Italia, ottenendo una risoluzione di 2^{-27} (tra 10^{-8} e 10^{-9} metri).
Pertanto, nel contesto delle coordinate del Bounding Box UTM32/WGS84 dell'Italia la rappresentazione FP64 ha le seguenti caratteristiche

<Base(FP64) =2, Risoluzione(FP64)=27>

2.3 Risoluzione di riferimento

In molti sistemi GIS utilizzabili per produrre o modificare dati geografici si permette/richiede all'utente di specificare la risoluzione delle coordinate in forma di risoluzione decimale.
Per evitare confusione, chiamiamo ***risoluzione di riferimento decimale*** RRD(R_i) la risoluzione specificata dall'utente in una rappresentazione R_i e ***griglia di riferimento*** la relativa griglia.

2.3.1 Problema nell'interpretazione di una risoluzione di riferimento decimale su una rappresentazione binaria.

Dato che molto spesso la rappresentazione reale è binaria, non è affatto ovvio interpretare il significato di una specificazione di risoluzione decimale su una rappresentazione binaria. La difficoltà è esemplificata dalla seguente figura 2.1.

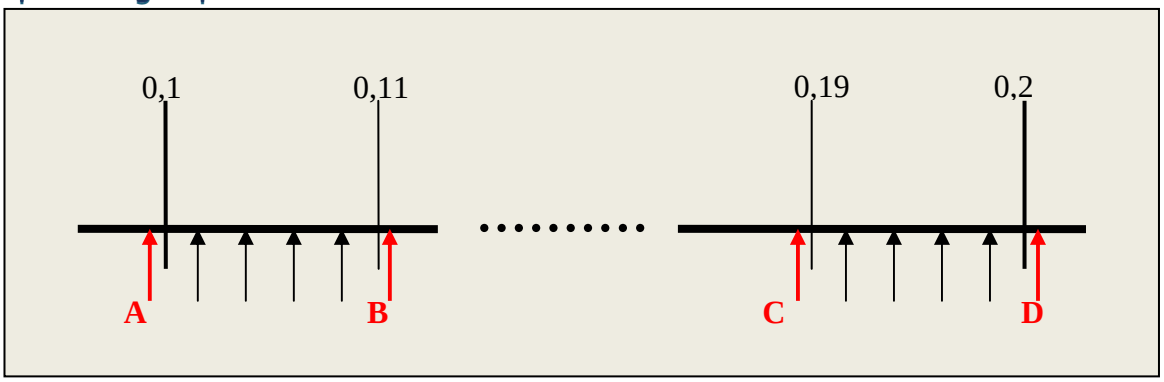


Figura 2.1

In questa figura le linee verticali indicano una griglia decimale con risoluzione 10^{-2} nell'intervallo 0,1-0,2 e le frecce indicano un'ipotetica griglia binaria; come si vede, nonostante la maggiore densità dei numeri della griglia binaria rappresentata, i numeri 0,1 - 0,11 - ... 0,19 - 0,2 non coincidono con nessun valore della griglia binaria (0,1 in rappresentazione binaria richiederebbe infiniti bit, perché è un numero periodico).
Se la rappresentazione reale è binaria e quindi la griglia reale è binaria, è errato immaginare la specificazione di una RRD come definizione di una griglia decimale; in realtà tale specificazione porta a una rappresentazione dei valori della (ipotetica) griglia decimale tramite il numero binario più vicino – le frecce rosse A, B, C e D di figura 2.1.
Per capire gli effetti particolari di questa situazione consideriamo il seguente esempio, con riferimento alla figura 2.1:

- supponiamo di utilizzare inizialmente una rappresentazione binaria R_i e una RRD=2 (cioè due cifre decimali)
  - → il numero 0,11 sarà rappresentato dal valore binario B di figura 2.1
- supponiamo poi di specificare un nuovo valore di RRD=1, cioè di chiedere di arrotondare a una sola cifra decimale
  - → nel mondo delle rappresentazioni decimali ci aspettiamo di approssimare 0,11 con 0,1
  - → ciò che in realtà accade nella rappresentazione binaria è la sostituzione del valore B con il valore A, cioè non è variato affatto il numero di bit di precisione interna utilizzati

Più paradossale è il seguente esempio:

- consideriamo la rappresentazione del numero 0,125 con RRD=3
  - → la rappresentazione binaria di questo numero è esatta, perché $0{,}125=2^{-3}$ (in rappresentazione binaria quindi si ha 0,001000…)
- se passiamo a RRD=1, cioè arrotondiamo il numero a una sola cifra decimale, otteniamo 0,1
  - → la rappresentazione binaria di 0,1 è periodica, quindi occupa tutti i bit disponibili e non è esatta

Pertanto, mentre in una rappresentazione decimale il passaggio da una rappresentazione a 2 cifre a una rappresentazione a 1 cifra riduce il numero di cifre necessarie alla rappresentazione***, nella rappresentazione binaria tale trasformazione non riduce affatto il numero di bit necessari***

2.3.2 Rappresentazione Binaria Approssimata

Abbiamo visto che la specificazione di una Risoluzione di Riferimento Decimale RRD(R_i)=k in un sistema che utilizza una rappresentazione binaria R_i non produce una griglia decimale, ma la selezione di un sottoinsieme della griglia di R_i in modo da ottenere una risoluzione ***circa*** uguale a 10^{-k}.
Chiameremo la rappresentazione che risulta dall'applicazione di RRD(R_i)=k la ***Rappresentazione Binaria Approssimata*** RBA(R_i, k).
Nota Bene. L'intervallo tra due numeri in RBA(R_i,k) può differire dal valore 10^{-k} perché sono presenti arrotondamenti di entità diversa in eccesso oppure in difetto; queste differenze sono ovviamente inferiori a Risoluzione(R_i), e quindi la griglia RBA(R_i,k) è tanto più regolare, quanto più la risoluzione binaria è piccola rispetto a 10^{-k}.

2.3.3 Limite su RRD.

Una proprietà che deve essere garantita nell'applicare una RRD(R_i)=k a una rappresentazione binaria R_i è la non ambiguità: per ogni numero della griglia di R_i deve esistere un solo numero decimale di k cifre corrispondente. In base a quanto detto sopra, ipotizziamo nel seguito che *i valori di k scelti soddisfino sempre la seguente relazione*

$$\textbf{\textit{risoluzione}}(R_i) < 10^{-(k+1)}$$

In altri termini, si richiede che la risoluzione interna della rappresentazione binaria R_i sia almeno 10 volte più fine di quella specificata come risoluzione di riferimento.

2.4 Applicazione al contesto di validazione dei dati

2.4.1 Caratteristiche dei sistemi e del data product

Sistema Sorgente e del Data Product

Il sistema sorgente ha una sua rappresentazione R_S e può applicare diverse RRD(R_S) nella produzione del Data Product DP con rappresentazione R_{DP}. Nel seguito ipotizziamo che le diverse operazioni svolte nel contesto sorgente, inclusa la produzione finale effettiva di DP permetta di caratterizzare quest'ultimo in base ai seguenti parametri:

- Base(R_{DP}): dipende dalla tecnologia scelta per il Modello Implementativo di interscambio; in particolare:
  - se tecnologia Shape: Base(R_{DP})=2
  - se tecnologia GML: Base(R_{DP})=10
- Risoluzione(R_{DP}):
  - se tecnologia Shape: Risoluzione(R_{DP})=9 (si veda quanto detto rispetto alla rappresentazione FP64 sul BB dell'Italia)
  - se tecnologia GML: da definire, in quanto il GML permette di utilizzare una stringa decimale di lunghezza variabile
- RRD(R_{DP}) questo parametro può essere scelto rispettando i limiti imposti dal punto precedente; tale scelta è importante per diversi motivi che verranno illustrati più avanti.

GeoUML Validator

Nel contesto del GeoUML Validator la rappresentazione R_V è FP64, mentre non è presente il concetto di RRD, perché il Validator deve interpretare i dati, ma non li modifica. Quindi:

- Base(R_V)=2

- Risoluzione(R_V)=9 (come sopra: rappresentazione FP64 sul BB dell'Italia)
- RRD: il Validator non specifica una RRD diversa da R_V, quindi i dati letti dal data product mantengono la RRD(R_{DP}); le eventuali coordinate di punti calcolati dal Validator però possono avere risoluzione R_V.

Sistema Destinazione

I contesti dei sistemi destinazione possono essere molteplici; nei capitoli seguenti emergeranno alcune condizioni per permettere che i dati letti in questo sistema non differiscano, come interpretazione topologica, dall'interpretazione fornita dal Validator.

2.4.2 Trasferimento dei dati

Quando DP viene letto dal Validator oppure dal sistema Destinazione possono verificarsi delle modifiche di coordinate dovute alla diverse rappresentazioni. Le condizioni in cui ciò può verificarsi e l'entità delle modifiche sono le seguenti:

- lettura di DP nel Validator:
  - se il DP usa un MI Shape, non dovrebbero verificarsi modifiche, perché $R_{DP}=R_V$
  - se il DP usa il MI GML, sicuramente si verificheranno, a causa del cambiamento di base, spostamenti di coordinate nell'ordine di Risoluzione(R_V), cioè di 10^{-9}.
- lettura di DP nel sistema Destinazione
  - in questo caso possono verificarsi spostamenti, in dipendenza della eventuale variazione della base della rappresentazione; tali spostamenti saranno dei seguenti ordini di grandezza
    - spostamenti nell'ordine di risoluzione(R_D) se il sistema destinazione non usa una RRD
    - spostamenti dell'ordine di RRD(R_D) in caso contrario

Le regole per evitare che questi eventuali spostamenti comportino modifiche della topologia saranno analizzate dopo aver preso in considerazione tutte le sorgenti di possibili problemi.

3. RAPPRESENTAZIONE VETTORIALE DELLA GEOMETRIA

Nelle implementazioni attuali, che seguono generalmente lo ***standard ISO Simple Feature Model (SFM)*** con estensioni per supportare il 3D, le geometrie sono rappresentate utilizzando un ***Modello Vettoriale*** in uno ***Spazio Discreto***, cioè uno spazio nel quale le coordinate dei punti sono numeri dotati di una precisione finita.
Il modello vettoriale costruisce le Geometrie secondo le seguenti regole:

1. Esistono delle primitive fondamentali dette ***Vertici***; un vertice è un punto definito da 2 o 3 coordinate (in base alle dimensioni dello spazio di riferimento); un vertice può essere isolato (in tal caso rappresenta una geometria puntiforme) oppure appartenere a una primitiva lineare (definita sotto)
2. Una ***Coordinata*** è un numero, rappresentato nel calcolatore in maniera finita secondo regole che dipendono dalla tecnologia adottata;
3. Utilizzando un certo numero di vertici e una regola di interpolazione è possibile costruire delle primitive lineari; per semplicità ci limitiamo qui a considerare delle primitive lineari rappresentate da due vertici con interpolazione lineare, dette ***Segmenti*** (di retta).
4. Tutti i tipi di geometrie lineari (***Curve***) sono ottenuti concatenando in maniera opportuna un certo numero di segmenti; le Curve implementate in questo modo sono generalmente dette ***Linestring***.
5. Tutti i tipi di geometrie areali (***Surface***) nel piano 2D sono rappresentate tramite un certo numero di anelli, cioè di curve chiuse, che ne rappresentano le frontiere esterne e interne (buchi) opportunamente strutturate; le superfici implementate in questo modo sono generalmente dette ***Polygon***.

ESEMPIO
In Figura 3.1 e nelle successive viene mostrata una griglia bidimensionale contenente geometrie vettoriali costruite secondo le regole definite.

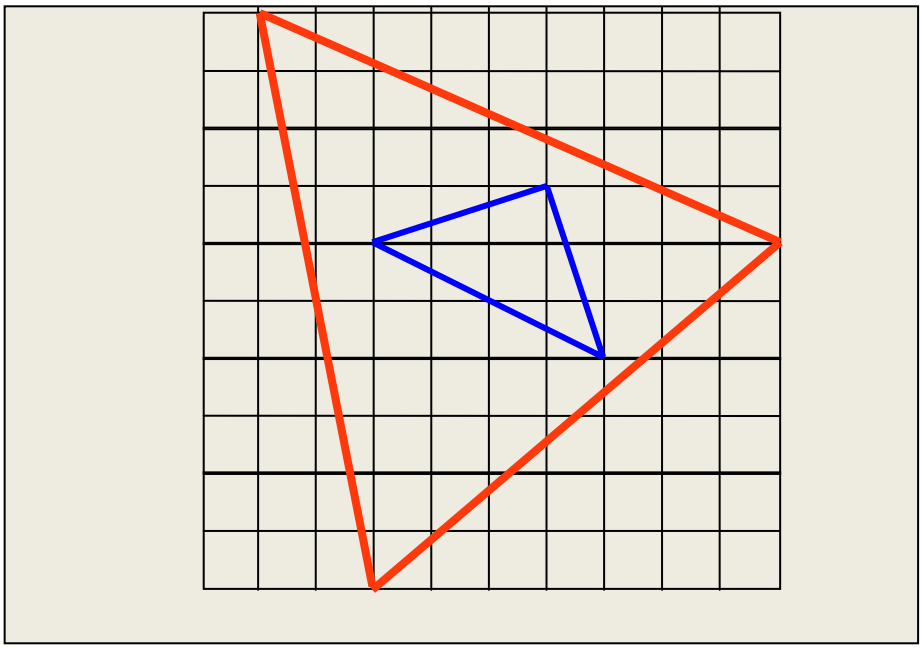


Figura 3.1

La griglia mostrata è generalmente quella sulla quale devono attestarsi tutti i vertici esplicitamente contenuti nelle geometrie. In base al contesto tale griglia può essere quella relativa alla rappresentazione interna R_i oppure alla Rappresentazione Binaria Approssimata RBA(R_i,k); in quest'ultimo caso è necessario tenere presente che il sistema è in grado di calcolare anche coordinate molto più fini della griglia mostrata.
Ad esempio, se il Validator legge un Data Product con RRD(R_{DP})=5 (cioè una risoluzione approssimata di 10^{-5}), tali dati possono essere rappresentati sulla griglia RBA(R_V,5), ma la risoluzione interna effettiva è Risoluzione(R_V), che, come già detto, è circa 10^{-9}. Pertanto il Validator può calcolare valori con precisione superiore di 4 ordini di grandezza.

4. PROBLEMI DI ELABORAZIONE DEI DATI

Data una rappresentazione vettoriale discreta di un insieme di geometrie, la valutazione delle proprietà topologiche presenta dei problemi.

4.1 Problemi dovuti all'approssimazione nei calcoli di coordinate – Tolleranza di Calcolo

La trasposizione su FP64 di algoritmi di geometria computazionale, pensati per funzionare nello spazio Euclideo con infiniti punti con coordinate reali, comporta il rischio di errori anche gravi nella valutazione dei predicati se un punto si trova a una distanza molto piccola da un segmento.
E' dimostrato che in tal caso non solo le distanze molto piccole possono essere rese equivalenti a 0, ma anche che è possibile una inversione di segno (un punto a destra è valutato a sinistra o viceversa).
Per limitare gli effetti di questi problemi il GeoUML Validator utilizza una **Tolleranza di Calcolo T_C** molto piccola nella valutazione dei predicati di uguaglianza: due coordinate che differiscono tra loro di un valore inferiore a T_C sono considerate uguali. La tolleranza di calcolo è molto piccola perché il suo scopo è solamente quello di eliminare gli effetti di arrotondamento numerico negli algoritmi e non di risolvere altri problemi (come discusso nel capitolo 5).
In figura 4.1 è mostrata una situazione in cui la tolleranza di calcolo può generare due diverse interpretazioni topologiche: se la distanza tra il vertice superiore del triangolo piccolo e il segmento del triangolo grande è valutata nulla, il triangolo piccolo è contenuto nel triangolo grande, altrimenti no.

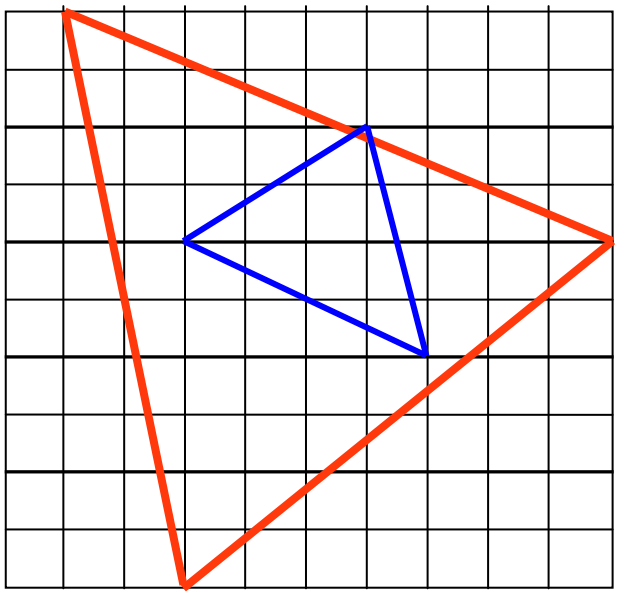


Figura 4.1

Questa figura mette in luce anche un importante caratteristica della rappresentazione vettoriale: mentre è possibile asserire che la distanza tra 2 punti è sempre maggiore della risoluzione di riferimento decimale RRD, la ***distanza tra un punto e un segmento può essere arbitrariamente piccola*** (in alcuni casi anche nulla) indipendentemente dalla risoluzione di riferimento.

4.2 Problemi dovuti al calcolo di intersezioni

La figura 4.2 mostra come la creazione di vertici in corrispondenza dei punti di intersezione tra 2 segmenti comporti, su una griglia finita, uno spostamento delle curve rappresentate.

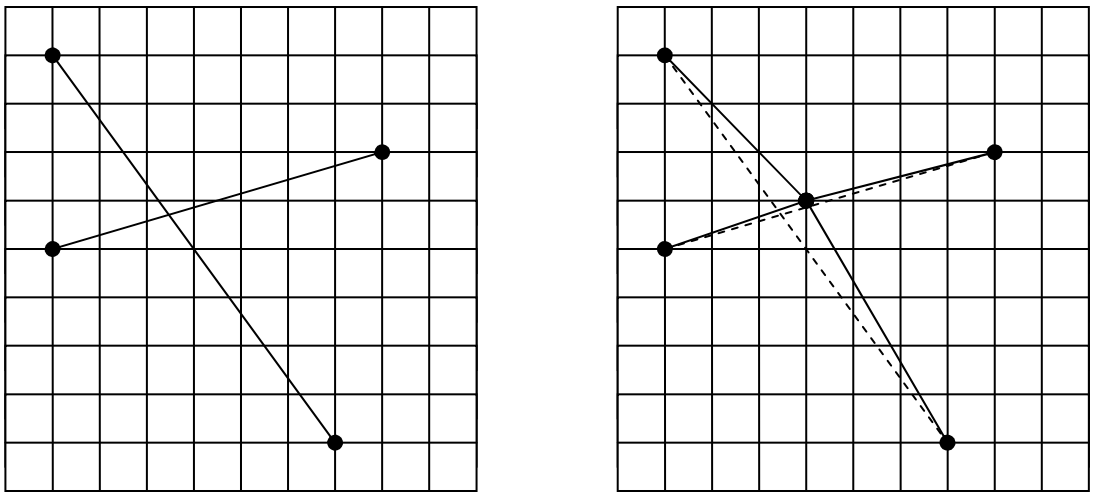


Figura 4.2

Questi spostamenti possono causare un cambiamento delle proprietà topologiche. Ad esempio, nella figura 4.3 si mostra che il punto A ha cambiato lato rispetto al segmento, quindi può avere cambiato una relazione topologica (ad esempio, se il segmento è un lato di un poligono, il punto passa da In a DJ o viceversa).

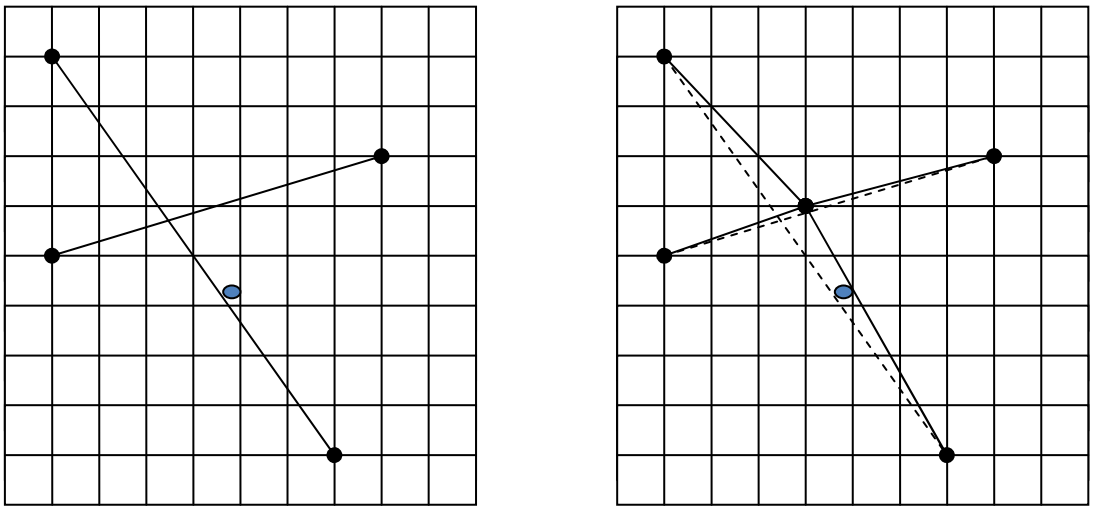


Figura 4.3

Nel contesto del Validator il calcolo di intersezioni avviene solamente per calcolare l'unione di geometrie nella valutazione di alcuni tipi di vincoli.

5. REGOLE PER EVITARE L'AMBIGUITA' TOPOLOGICA

A causa dei problemi evidenziati nei capitoli precedenti la valutazione delle proprietà topologiche è soggetta a un'incertezza che può condurre a ambiguità topologica; in questo capitolo si propongono delle regole per eliminare tale ambiguità.

5.1 Determinazione dell'incertezza e regole per ridurla

L'incertezza topologica è primariamente dovuta all'incertezza nella valutazione della posizione di un punto rispetto a un segmento. Possiamo rappresentare l'incertezza come un ***Buffer di Incertezza*** di larghezza **I** attorno ad ogni segmento: i punti situati all'interno di tale buffer sono posizionati in maniera ambigua rispetto al segmento.

Dalle analisi precedenti la larghezza del Buffer di Incertezza risulta dipendere dai seguenti elementi:

- il cambiamento delle rappresentazioni nel trasferimento dei dati, e in particolare le risoluzioni applicate in DP, V e D
- gli spostamenti dovuti ad elaborazioni nella valutazione di relazioni topologiche
- gli effetti dovuti agli algoritmi di calcolo – questi ultimi effetti sono normalmente più piccoli dei precedenti

Per tenere limitata la larghezza del buffer di incertezza è opportuno seguire le regole seguenti:

Regola relative alla risoluzione:

- Regola di **conservazione della risoluzione**: Non ridurre la risoluzione di riferimento nel trasferimento dei dati

In un contesto in cui i dati devono essere trasferiti avanti e indietro tra sistemi cooperanti è opportuno sostituire questa regola con la seguente più stringente:

- Deve esistere una risoluzione di riferimento comune a tutti i sistemi cooperanti

Regola relative alla Modalità di Valutazione delle proprietà topologiche:

- Evitare il più possibile di dover calcolare nuove intersezioni nella valutazione delle proprietà topologiche
- Nel calcolo di nuovi punti (tipicamente intersezioni) utilizzare la massima risoluzione possibile

L'applicazione di queste regole non elimina l'incertezza, ma mantiene limitata la larghezza I del buffer di incertezza.
In un contesto nel quale sono stabilite le caratteristiche delle varie rappresentazioni utilizzate possiamo ipotizzare che sia nota tale larghezza; per un tale contesto possiamo definire delle regole di strutturazione del data product che eliminino l'ambiguità topologica anche in presenza di tale incertezza.

5.2 Regole di strutturazione dei dati in presenza di incertezza

Le seguenti due regole caratterizzano il modo in cui deve essere strutturato un Data Product DP per permettere una valutazione non ambigua delle sue proprietà topologiche in presenza di un certo grado di incertezza I:

1. Regola di **coincidenza esatta**: ogni punto che deve appartenere a un segmento deve coincidere (avere coordinate identiche) con un vertice del segmento
2. Regola di **distanza minima**: la distanza tra ogni punto e ogni segmento al quale il punto non deve appartenere deve essere superiore a una distanza minima $\boldsymbol{\delta_{min}}$ **> 10*I**

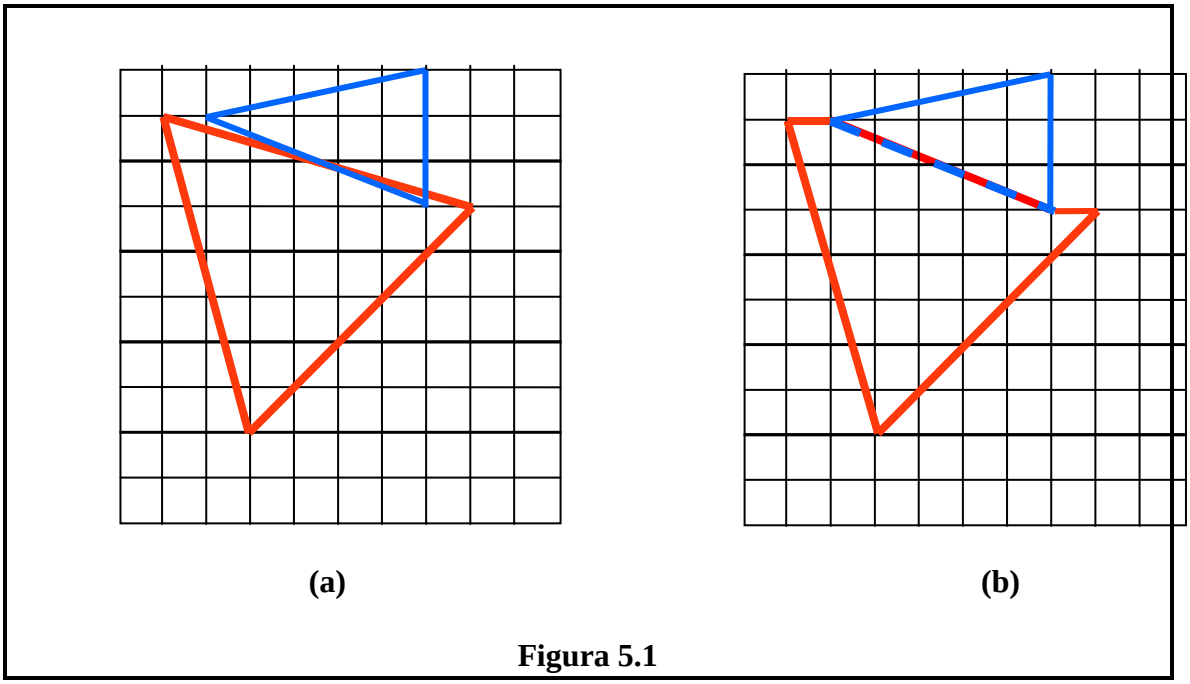
Esempi di Incertezza e di Distanza Minima:

- lettura di un DP in formato Shape nel Validator: in questo caso l'incertezza è minima, perché le due rappresentazioni sono uguali, non ci sono spostamenti in lettura, gli spostamenti per calcolo oppure per creazione di intersezioni sono dell'ordine di risoluzione(R_V); quindi I= R_V → $\boldsymbol{\delta_{min}}$ = 10* R_V = 10^{-7} è sufficiente
- lettura di un DP in un sistema destinazione con RRD(R_D)=5 e variazioni di rappresentazione; in questo caso l'incertezza dovuta a spostamenti possibili è dell'ordine di R_D , cioè 10^{-5} ,e quindi è opportuna una distanza minima $\boldsymbol{\delta_{min}}$ = 10* R_D = 10^{-4}

Nel seguito vediamo alcuni esempi di applicazione delle due regole e poi procediamo a una discussione del perché tali regole funzionano e non sono sostituibili dal tradizionale concetto di tolleranza.

Esempio regola 1 (coincidenza esatta)

Se si vuole che due poligoni siano adiacenti, non basta procedere come in figura 5.1a ma è necessario inserire 2 vertici nel poligono rosso come in figura 5.1b.
Nel caso di figura 5.1.a l'interpretazione della relazione esistente tra i 2 poligoni può risultare di adiacenza (TC) oppure di sovrapposizione (OV) in base al fatto che la distanza tra i vertici del triangolo piccolo e il lato del triangolo grande sia interpretata come 0 o maggiore di 0, e quindi dipende dall'impiego o meno di una tolleranza di calcolo e dal suo valore.



(a) **(b)**

Figura 5.1

Esempio regola 2

Se interpretiamo la seguente figura 5.2 come un poligono bucato, allora la distanza tra i vertici del triangolo interno e i lati del triangolo esterno deve essere maggiore di δ_{min} per garantire la proprietà di contenimento dell'anello interno nell'anello esterno.

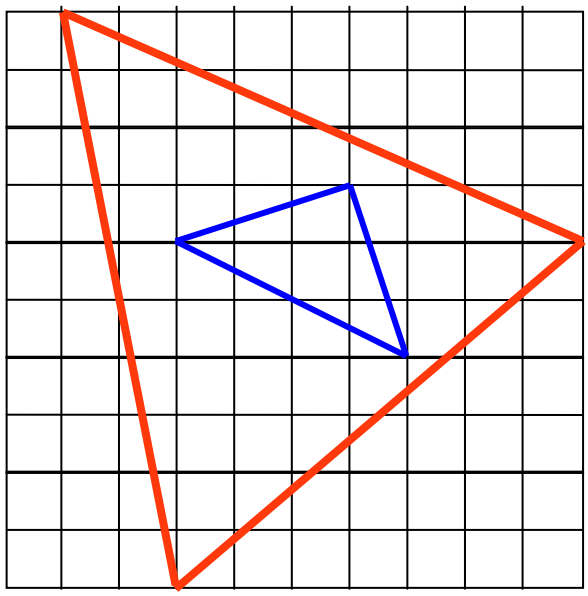


Figura 5.2

5.3 Discussione delle regole relative alla rappresentazione dei dati

Le regole relative alla rappresentazione dei dati (coincidenza esatta + distanza minima) spesso non sono applicate, ritenendo che possano essere sostituite da un uso opportuno del concetto di tolleranza.

5.3.1 Distinzione tra Valutazione e Creazione della correttezza topologica

E' opportuno tenere presente che lo scopo del Validator è di ***valutare*** la correttezza dei dati senza modificarli.
La valutazione della correttezza topologica è da tenere ben distinta dalle operazioni di "***editing topologico***", che mirano a ***creare*** la correttezza topologica modificando i dati.
E' opportuno tenere presente che ***la valutazione delle proprietà topologiche è fatta non solo dal Validator ma implicitamente da tutte le applicazioni che accedono i dati utilizzandone le proprietà topologiche (comprese le interrogazioni topologiche)***.
E' pertanto del tutto sensato richiedere che l'editing topologico iniziale di un dato miri a soddisfare tutte le regole che permettono un uso semplice e non ambiguo da parte delle applicazioni che lo utilizzano.

L'uso di una tolleranza T è molto diverso nei due casi:

- nell'editing topologico
  - due punti sono fusi, spostandoli, se la loro distanza è minore di T
  - un punto posizionato a distanza inferiore a T da un segmento viene inserito come vertice nel segmento
- nella valutazione delle proprietà topologiche
  - due punti vengono considerati uguali se la loro distanza è minore di T (ma rimangono inalterati)
  - un punto viene considerato appartenente a un segmento se è posizionato a distanza dal segmento inferiore a T

Nella sezione 5.3.2 si mostra che, ai fini della ***valutazione*** non ambigua della correttezza topologica di un Data Product, il concetto di tolleranza non è sufficiente.
L'uso di algoritmi basati sulla tolleranza (insieme ad altri accorgimenti, come lo snapping, ecc…) è invece sicuramente utile nell'editing topologico mirante a creare un Data Product che possiede le caratteristiche adatte per supportare la valutazione non ambigua delle sue proprietà topologiche.

Inoltre, ***il concetto di tolleranza soffre di una carenza di definizione per quanto riguarda la sua applicazione ricorsiva*** (cioè quando molti punti sono vicini l'uno all'altro), per cui i diversi sistemi che usano tale concetto possono avere comportamenti diversi. Anche per questo motivo è opportuno lasciare la responsabilità di verificare gli effetti dell'impiego della tolleranza a chi crea il dato e opera in un sistema ben noto, ma poi utilizzare le proprietà topologiche con regole indipendenti dal sistema che le utilizza.

5.3.2 Confronto della valutazione basata sulla tolleranza con quella basata su coincidenza esatta + distanza minima

La figura 5.3 mostra un segmento (linea continua), una linea tratteggiata LD posta a distanza D dal segmento, e 2 gruppi di punti:

- ***punti lontani***: i punti rappresentati da cerchi neri sono vicini alla linea tratteggiata, sono quindi a una distanza dal segmento circa uguale a D
- ***punti vicini***: i punti rappresentati da rombi rossi sono vicini al segmento.

La griglia mostra la risoluzione (R_V se stiamo considerando il Validator).

Il rettangolo ombreggiato, di larghezza **I** circa uguale alla risoluzione R_V, rappresenta il buffer di incertezza relativo a LD.

Applicazione del concetto di tolleranza:
Applicando le regole relative alla tolleranza ai punti di figura 5.3, nella situazione critica in cui D=T, otteniamo:

- i punti vicini sono considerati appartenenti al segmento (non c'è ambiguità)
- i punti lontani sono in situazione ambigua, perché piccole variazioni (dell'ordine della risoluzione interna) possono far variare la loro interpretazione da punti che appartengono al segmento a punti che non gli appartengono

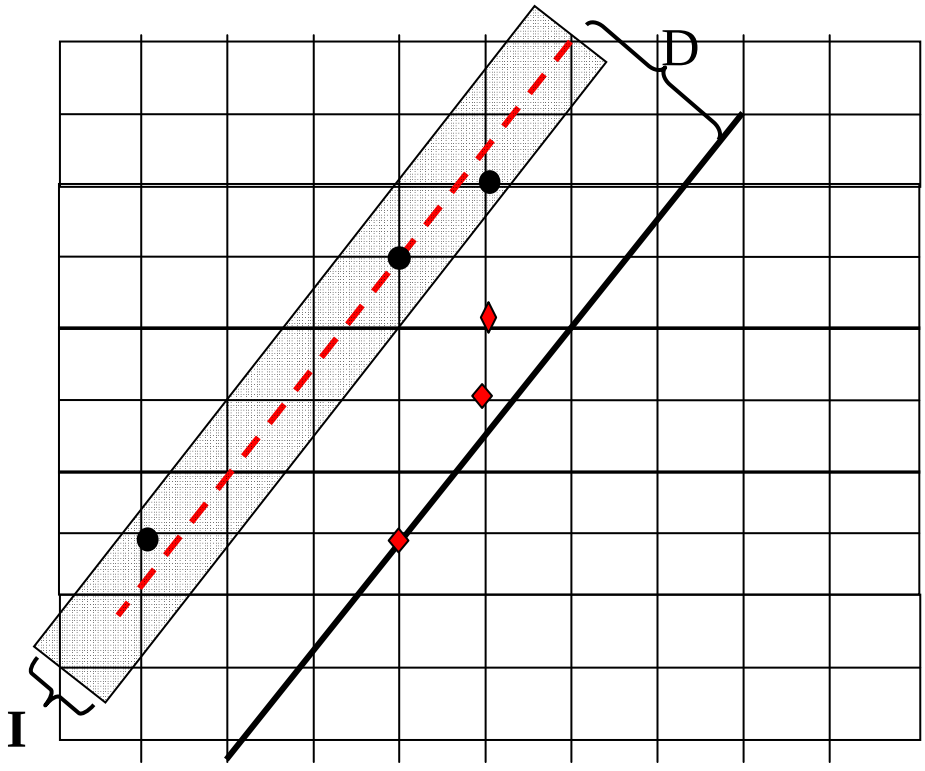


Figura 5.3

Applicazione delle regole di coincidenza esatta + distanza minima:
Utilizzando queste regole, nella situazione critica in cui sia D= $\boldsymbol{\delta_{min}}$ otteniamo:

- i punti vicini violano le regole; è necessario inserire dei vertici per correggere la situazione, oppure spostarli alla distanza minima
- i punti lontani sono considerati tutti separati dal segmento, indipendentemente da piccole variazioni (infatti: i punti a distanza maggiore o uguale a (δmin – I/2)

sono considerati corretti ed esterni al segmento; i punti a distanza inferiore a (δmin – I/2) devono essere posizionati sul segmento)

La figura mostra che le regole permettono di separare nettamente i punti che appartengono a un segmento da quelli che non gli appartengono.

Precisazione

Le regole di coincidenza esatta e di distanza minima al momento possono costituire solamente una raccomandazione, in quanto il Validator non le controlla esplicitamente.
In particolare, il Validator considera appartenente a un segmento un punto che sia giudicato a distanza nulla in base alla tolleranza di calcolo, ma tale tolleranza è molto piccola.
E' possibile che un data product contenga punti molto vicini ai segmenti, ma privi di un vertice corrispondente. In questo caso si potrebbero ottenere degli errori topologici dovuti all'incertezza oppure l'assenza di errori che poi invece emergono nel sistema destinazione.

5.4 Applicazione delle regole di coincidenza esatta e di distanza minima ai vincoli GeoUML

Il GeoUML definisce a livello concettuale alcune proprietà topologiche che devono essere soddisfatte da DP; tali proprietà sono definite in due aspetti del GeoUML:

- nei vincoli topologici e di composizione e
- nelle proprietà topologiche dei tipi geometrici (ad esempio, la frontiera interna di un poligono deve essere contenuta nella frontiera esterna).

Le regole di coincidenza esatta e di distanza minima si applicano sia a vincoli espressi nello spazio 2D, sia a vincoli espressi nello spazio 3D.

5.4.1 Vincoli che richiedono Coincidenza esatta (senza calcolo di unioni)

Coincidenza esatta tra geometrie di interi oggetti

- Vincoli *esiste* e *perOgni* con relazione EQ (identità tra geometrie PT-PT, L-L)

Coincidenza esatta tra porzioni di geometrie di singoli oggetti

- Vincoli *esiste* e *perOgni* con relazione TC
  - tra poligoni (identità L-L su tratti di frontiera)
  - tra curve (identità PT-PT su curve)
  - tra poligoni e curve (identità L-L su tratti di frontiera)
  - tra punti e poligoni (identità Pt-Pt con vertici della frontiera)
  - tra punti e curve (identità Pt-Pt con vertici estremi della curva)
- Vincoli *esiste* e *perOgni* con relazione IN
  - tra curve (identità L-L su curve)
  - tra punti e curve (identità Pt-Pt con vertici interni della curva)

Per garantire il soddisfacimento di questi vincoli DP deve contenere tutte le geometrie condivise all'interno degli oggetti. Nel processo di produzione queste devono essere determinate e inserite negli oggetti.

5.4.2 Vincoli che richiedono Coincidenza esatta e calcolo di unioni

- Vincolo *unione* con relazione EQ (identità tra oggetti e oggetti derivati per unione)

- Vincoli *CompostoDa* e *Partizionato* (identità tra oggetti e oggetti ricostruiti per unione)

Anche per questi vincoli DP deve contenere tutte le geometrie condivise all'interno degli oggetti. Nel processo di produzione questi devono essere determinate e inserite negli oggetti.
Questa operazione richiede di precalcolare le unioni e di inserire gli eventuali vertici creati da tali operazioni nelle geometrie. Pertanto, in base all'ipotesi precedente, l'esecuzione di un'unione da parte del validatore non altera i dati.

Esempio
Sia C *compostoda* A e B.
In produzione: C viene creato con l'operazione C= unione A,B ed esplicitamente memorizzato
Verifica del vincolo nel validatore:

1. crea D= unione A,B
2. verifica che C=D

Se nell'operazione 1 vengono creati dei vertici la verifica 2 può risultare falsa, perchè la costruzione dell'unione nel validatore può produrre tali vertici in maniera diversa che nel produttore. Le Regole Topologiche di Implementazione dicono che il produttore deve creare tutti i vertici dovuti alle intersezioni in fase di unione e riportarli in A e B. In questo caso il validatore non dovrebbe creare vertici.

5.4.3 Vincoli che richiedono Distanza minima (senza calcolo di unioni)

- Vincoli *esiste* e *perOgni* con relazione DJ e OV tra qualsiasi tipo geometrico
- Vincoli *esiste* e *perOgni* con relazione IN tra poligoni, linee/poligoni, punti/poligoni

5.4.4 Vincoli che richiedono Distanza minima (con calcolo di unioni)

- Vincolo C (RelTopo) unione A,B con.
  - RelTopo è una relazione topologica escluso EQ (eguaglianza)
  - A, B e C sono classi poligonali

Verifica del vincolo:

1. crea D= unione A,B
2. verifica C (RelTopo) D

5.4.5 Osservazione sui vincoli su geometrie 2D derivate da planar di geometrie 3D

Dal punto di vista della valutazione dei vincoli si applicano le regole del 2D e quindi non ci sono problemi.
In produzione: Per quanto riguarda la regola di coincidenza esatta esiste un problema: come riportare sulle geometrie 3D i vertici generati per intersezioni sulle rispettive proiezioni planari.

Esempio: Supponiamo di avere un vincolo "Point2D (IN) *esiste* Curve3D.*planar*"

In base alla regola di coincidenza esatta dovrebbe esistere un vertice sulla Curva le cui coordinate planari X e Y sono identiche a quelle del punto. Dato che la curva è 3D, in

produzione è necessario determinare per interpolazione un valore della coordinata Z da associare a tale vertice.