Progettazione di architetture software

Indice

9. Progettazione di architetture software

> *For the most part things never get built the way they were drawn. It is a small miracle that the memorial looks exactly the way it was designed.*
> Maya Lin , *Academy of Achievement Interview,*
> June 16, 2000, Scottsdale, Arizona. [1]

Architettura software

astrazione del come si compone ed organizza il sistema software

In questa lezione:

- **introduzione all'architettura software come:**
  - **disciplina**
  - **attività e prodotto di progettazione**
- **panoramica di stili di architetture software**

9.1 Introduzione

oltre 90 definizioni di *architettura software*
al sito del *Software Engineering Institute,* CMU (SEI)

9.1.1 L'architettura nel progetto software

- **progetto dell'architettura: fase dello sviluppo, ma...**
- **l'astrazione architetturale è iterabile a parti del sistema**
- **utilità delle architetture software:**
  - **riconoscimento di tratti comuni a sistemi diversi**
  - **supporto ad analisi, valutazione di alternative, decisioni**
  - **riusabilità per implementazioni differenti**
  - **documentazione ad un livello di astrazione appropriato**

9.1.2 Origini della disciplina

alcuni riferimenti:
(Parnas 1972), (Perry et al. 1992), (Garlan et al. 1993), (Shaw et al. 1996), (Shaw et al. 1997)

- **crescita progressiva del livello di astrazione concettuale nella storia dei linguaggi di programmazione**
- **parallela crescita dell'interoperabilità di componenti software eterogenei nella storia**

dei sistemi informatici
- **la crisi del software (fine anni '60)**
- **sviluppo della teoria dei tipi di dati astratti (ADT) (specifica algebrica, deduzione automatica, CASL, etc., dagli anni '70)**
- **sviluppo di architetture e modelli di sistemi software (ISO/OSI RM, NIST/ECMA RM, OMG/CORBA, etc., dagli anni '80)**

9.2 Caratteristiche di tipi di architetture software

altrimenti detti stili di architetture software

- **parti** (*componenti*)
  - *e.g.* filtri, classi, oggetti, strati, ...
- **relazioni** fra parti
  - *e.g.* ereditarietà, *client/server* , adiacenza, ...
- **interazioni** (*connettori*)
  - *e.g.* canali, metodi, servizi, ...
- **vincoli** sulla combinabilità di parti e/o interazioni
  - *e.g.* vincoli topologici, di cardinalità, ...

9.2.1 Architetture dataflow

altrimenti dette a **canali** (*pipes*) e **filtri**

- *componenti:* **filtri** (trasformatori I/O su **flussi** (*stream*) di dati)
- *connettori:* **canali** (supporto di flussi sequenziali)
- *vincoli:* **località** delle trasformazioni

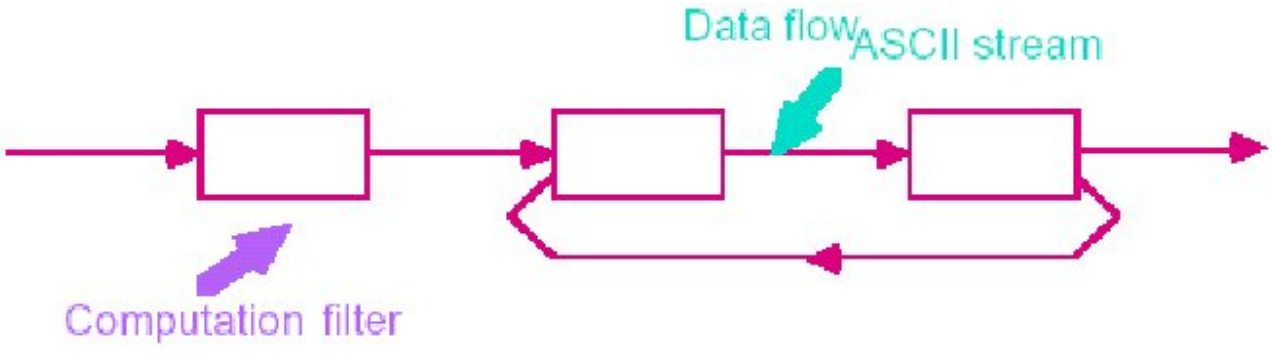


Figura 9.1: Filtri e canali (Garlan et al. 1993, p. 7)

casi speciali:
- architettura *pipeline* (topologia sequenziale)
- canali a **capacità limitata**
- canali **tipati**

9.2.2 Architetture ad oggetti

o di tipi di dati astratti (ADT)

- *componenti:* **oggetti** (istanze di classi, o di ADT)
- *connettori:* **metodi** (o operazioni, invocazioni di)
- *vincoli:* **incapsulamento** (accesso ai dati *solo* attraverso metodi)

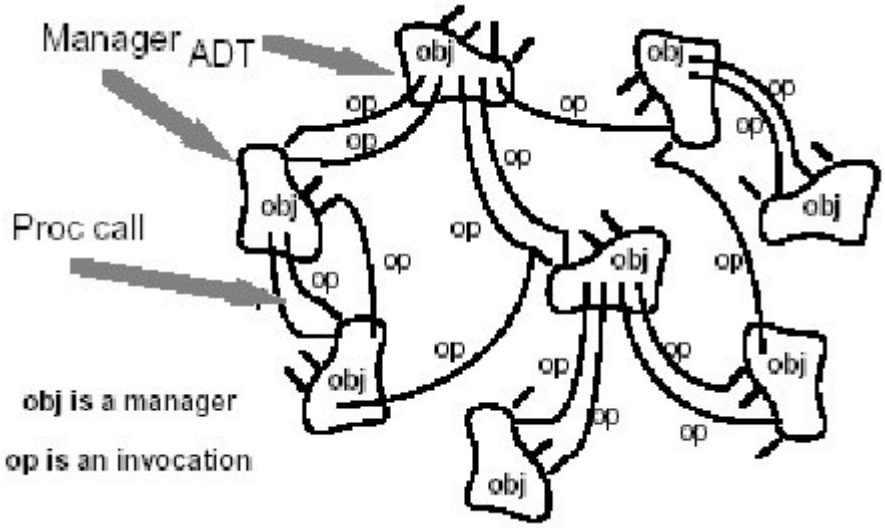


Figura 9.2: ADT e oggetti (Garlan et al. 1993, p. 8)

discussione

dove si situa l'ereditarietà?

9.2.3 Architetture basate su eventi

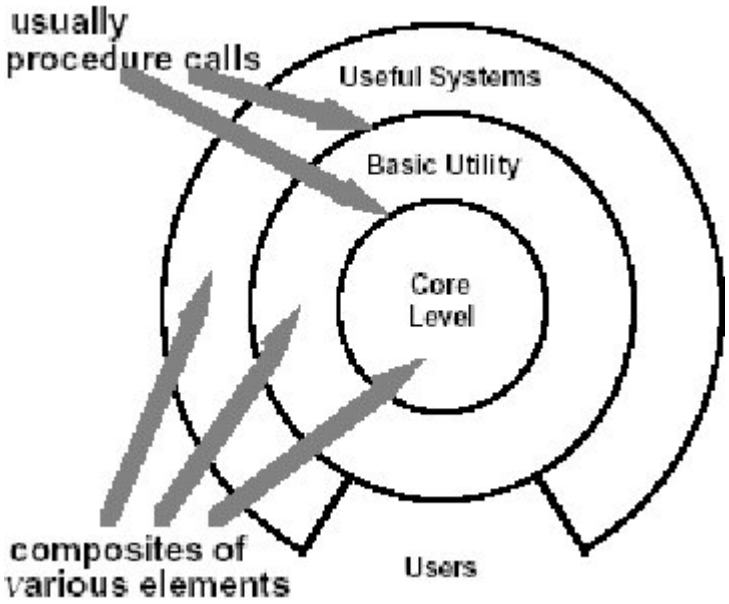
ad invocazione implicita

- *componenti:* **moduli** (dotati di interfacce)
- *connettori:* **associazioni** di procedure a eventi
- *vincoli:* **globalità** degli eventi, **località** delle associazioni
- *nota:* spesso, in queste architetture, l'invocazione **esplicita** è un meccanismo complementare di interazione

vantaggi: **riuso** di componenti, **adattabilità** dell'architettura
svantaggi: globalità del flusso del controllo, dipendenza della semantica delle procedure di annuncio di eventi dal contesto
difficoltà di analisi

9.2.4 Architetture stratificate

- *componenti:* **strati** (*layers*)
- *connettori:* **servizi** (primitive di, punti di accesso a)
- *vincoli:* **adiacenza** degli strati direttamente interagenti



Strati di software (Garlan et al. 1993, p. 11)

9.2.5 Architetture centrate su repository

- *componenti:* **repository** di dati, **moduli esterni**
- *connettori:* **servizi** del *repository* (primitive di, punti di accesso a)
- *vincoli:* **interazione** fra i moduli **mediata** dal *repository*
- *casi speciali:* **blackboard architecture** (ks = *knowledge sources*)

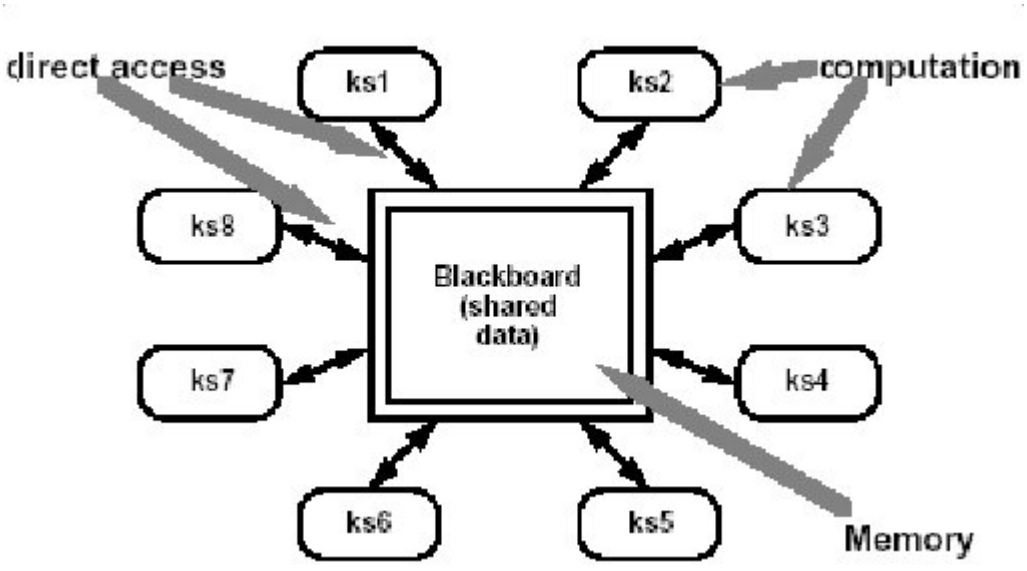


Figura 9.4: Blackboard (Garlan et al. 1993, p. 13)

9.2.6 Architetture basate su interprete

- *componenti:* **macchina virtuale + 3 strutture dati**
  (per la simulazione dell'esecuzione del programma interpretato,
  v. Fig. 9.5)
- *connettori:* **?** (v. Fig. 9.5)
- *vincoli:* **computazione table-driven**

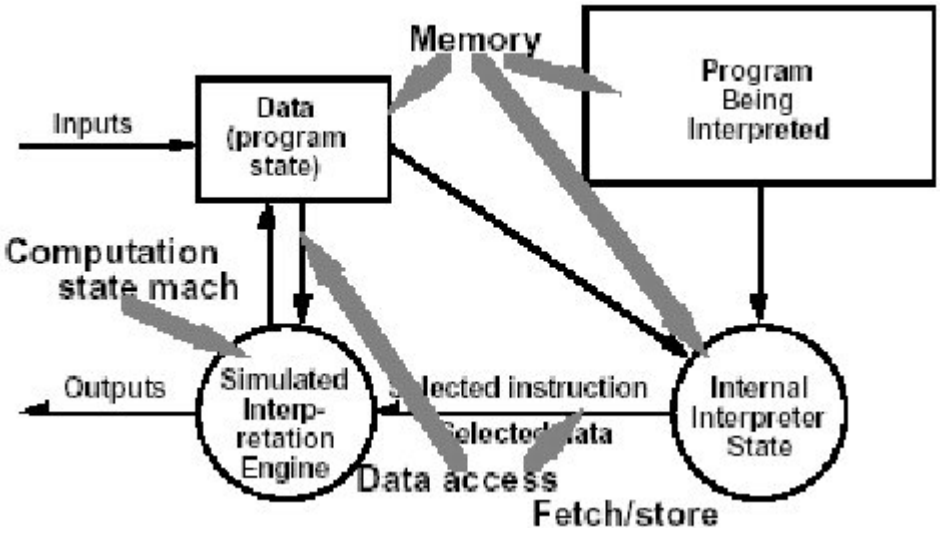


Figura 9.5: Interprete (Garlan et al. 1993, p. 14)

9.3 Altre architetture software

altri stili (?) di uso frequente:

- **architetture distribuite (client/server, RMI, CORBA, ...)**
- **architetture sequenziali main program / subroutines**
- **sistemi di transizioni di stati**
- **sistemi di controllo di processo**

9.3.1 Architetture software eterogenee

risultano da:

- **iterabilità dell'astrazione architetturale a parti (e connettori) del sistema, applicando stili diversi a parti diverse**
- **possibilità che un componente interagisca con parti diverse attraverso diversi tipi di connettori**
- **uso di stili diversi a diversi livelli della descrizione**

9.4 Esercizi

Tema

Approfondimenti di casi di studio

9.4.1 Approfondimenti di casi di studio

Per questa lezione si propongono come esercizi gli approfondimenti di casi di studio di architetture software, quali quelli presentati in Sez. 4 di (Garlan et al. 1993), o altri di cui si sia a conoscenza.

9.5 Note

1. Perlopiù le cose non vengono mai costruite nel modo in cui erano state tracciate. È un piccolo miracolo che il memoriale [dei veterani del Vietnam] appaia esattamente nel modo in cui è stato progettato.

9.6 Bibliografia

- **SEI**. *Software Engineering Institute, Carnegie Mellon University*. Web: http://www.sei.cmu.edu.
- **Parnas, D.L.**, 1972. On the criteria to be used in decomposing systems into modules. *Communications of the ACM*, 15:12, 1053-8.
  Web: http://www.acm.org/classics/may96
- **Perry, D.E. & Wolf, A.L.**, 1992. Foundations for the study of software architecture. *ACM SIGSOFT Softw. Eng. Notes*, 17:4, 40-52.
  Web: http://www.ece.utexas.edu/~perry/work/swa
- **Garlan, D. & Shaw, M.**, 1993. An Introduction to Software Architecture. *In:* **Ambriola, V. & Tortora, G.**, ed. *Advances in Software Engineering and Knowledge Engineering*. Singapore: World Scientific Publishing Company, 1-39.
  Web: http://www-2.cs.cmu.edu/afs/cs/project/able/ftp/intro_softarch/intro_softarch.pdf
- **Shaw, M. & Clements, P.**, 1997. A Field Guide to Boxology: Preliminary Classification of Architectural Styles for Software Systems (2 files). *In: Proc. 21st Int'l Computer Software and Applications Conference*. Pittsburgh, PA 15213: Carnegie Mellon University, 6-13.
  Web: http://spoke.compose.cs.cmu.edu/shaweb/p/pubs.htm
- **Shaw, M. & Garlan, D.**, 1996. *Software Architecture: Perspectives on an Emerging Discipline*. Prentice Hall.
  Web: http://www.prenhall.com/allbooks/esm_0131829572.html