Why Distributed Systems?

Distributed Systems

Andrea Omicini

mailto:andrea.omicini@unibo.it
Dipartimento di Informatica – Scienza e Ingegneria (DISI)
Alma Mater Studiorum – Università di Bologna

Academic Year 2025/2026

Pervasive Computation & Interaction I

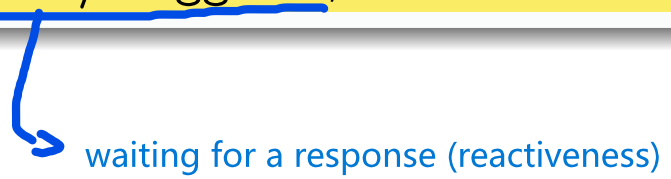
Nowadays, computational systems...

- ...have become pervasive, since they are everywhere, and tend to affect every aspect of our everyday life and activity
- ...are at the core of most (if not all) *artificial systems*, so that every principled discipline for modelling / engineering computational systems affects the *modelling and engineering of almost every sort of artificial systems*

Pervasive Computation & Interaction II

Pervasiveness of computations

- we live immersed in a sort of ever-expanding *computational bubble*, where an huge number of computations are performed at every instant around us
  - at home, in our cars, in the workplaces, in hospitals, in airports and train stations, in schools and universities, in public spaces, ...
- distributed, *concurrent* computations (they aren't the same thing)
  - either controlled / triggered, or autonomous computations



Pervasive Computation & Interaction III

Pervasiveness of interaction

- almost any computational system of today comes equipped with ICT technologies for interacting with other computational systems
- computational devices *continuously interact*
  - with humans
  - with each others
  - with the physical *environment* and its *resources*

Physical vs. Computational

The physical nature of artificial systems...

...adds complexity to computational components / systems

La fisicità introduce problemi che il software puro non ha:

- in terms of distribution
  - spatial *distribution*
  - temporal *distribution*

I componenti sono separati da una distanza fisica, il che implica latenza di comunicazione, possibili disconnessioni o interferenze.

La sincronizzazione dell'orologio tra i componenti fisicamente separati è difficile e le azioni avvengono in tempi reali non idealizzati.

- in terms of unpredictability of the *environment* where they have to work

Il mondo reale è caotico. I sistemi fisici devono gestire variazioni di temperatura, luce, umidità, interferenze elettromagnetiche, ostacoli in movimento e guasti meccanici, che un sistema puramente computazionale in un datacenter non affronta allo stesso modo.

On the Notion of Distribution I

What is *spatially* distributed?

1. *computational* units — Sono i processori o i server fisici. In un sistema distribuito, non sono tutti nello stesso datacenter, ma possono essere sparsi geograficamente
2. *communication* channels — I mezzi fisici o logici usati per lo scambio di messaggi sono distribuiti nello spazio, con conseguente latenza e possibili guasti.
3. *data / information / knowledge*
   - along with their *representations*

   L'informazione stessa è replicata o frammentata su diverse unità computazionali. Il modo in cui viene rappresentata è anch'esso distribuito.
4. sensors, actuators, ... — Questi elementi sono per natura sparsi. I sensori raccolgono dati da luoghi diversi e gli attuatori eseguono azioni in punti diversi dell'ambiente.
   - the boundary between the system and the surrounding *environment* is spatially *sparse*

Questo significa che, poiché il sistema è composto da tanti elementi sparsi che interagiscono con l'ambiente (sensori, attuatori), non c'è una singola "scatola" che lo racchiude. L'interfaccia tra il sistema artificiale e il mondo esterno (l'ambiente) non è concentrata, ma è distribuita su un ampio volume o area.

On the Notion of Distribution II

In un singolo computer, c'è un orologio centrale e gli eventi hanno una sequenza precisa e non ambigua. In un sistema distribuito, con molti orologi non perfettamente sincronizzati, è impossibile stabilire un ordine globale degli eventi.

What is *temporally* distributed?

- basically, events
  - things happens, yet no longer in a clearly-ordered sequence

ciò che si distribuisce non sono solo le unità fisiche, ma anche gli eventi, ovvero le azioni o le occorrenze che avvengono all'interno del sistema (es. un sensore che rileva un dato)

- events are scattered, and the trivial *before/after* temporal relation simply cannot be applied to many pairs of events

sparsi

banale relazione temporale "prima/dopo"

Se due eventi avvengono su due computer diversi, non possiamo dire con certezza quale sia avvenuto prima, a meno che non ci sia un messaggio che li connette.

Spatio-temporal unity of systems is lost

- there is no longer a notion of *system time*, nor a system *location*
- system components, at different level of abstraction, are only *partially correlated*
  - temporally & spatially

or sometimes

I vari pezzi del sistema (software, hardware, dati) hanno solo una correlazione parziale. Ad esempio, un sensore a Milano e un server a Tokyo sono correlati per il loro scopo comune, ma operano in tempi e spazi diversi. Questa mancanza di correlazione completa è ciò che rende i sistemi distribuiti così complessi da gestire.

Non c'è più una nozione singola e unificata di "tempo di sistema" né di "posizione del sistema". Il sistema non esiste in un solo punto dello spazio o del tempo, ma è un insieme di componenti che esistono in luoghi e momenti diversi.

On the Notion of Distribution III

la distribuzione costringe gli ingegneri a progettare sistemi che funzionino anche senza l'illusione di un tempo unico e senza la necessità che i componenti siano fisicamente o logicamente vicini.

Nei sistemi monolitici (un solo computer), tutti gli eventi (istruzioni eseguite) avvengono in una sequenza chiara e univoca: ordinamento totale. Poiché gli eventi avvengono su unità computazionali diverse e separate (senza un orologio centrale perfetto), non possiamo dire con certezza quale sia avvenuto prima tra due eventi non correlati. L'unica relazione che resta è l'ordinamento parziale: possiamo dire che l'evento A è avvenuto prima dell'evento B solo se A ha causato B (ad esempio, A è l'invio di un messaggio e B è la sua ricezione).

assunzioni — non sono più valide

A number of assumptions over systems no longer hold

1. system events *no longer* constitute a totally-ordered set
   - generally speaking, *partial ordering* is the only feature
2. admissible interactions among system components *no longer* depend on *compresence*

tra i componenti del sistema

compresenza ("Compresenza" si riferisce all'essere presenti nello stesso luogo e nello stesso momento)

   - in space / time
   - within the same physical / virtual *topology*

Nel sistema monolitico: Due moduli software interagiscono perché condividono la stessa memoria o lo stesso bus di sistema (sono "compresenti").
Nel sistema distribuito: I componenti interagiscono (si scambiano messaggi, aggiornano dati) anche se sono separati da migliaia di chilometri e operano in fusi orari diversi. L'interazione avviene in modo asincrono e remoto. Non è richiesta la vicinanza (spazio) né l'identità di tempo (tempo) o di ambiente (topologia) per comunicare.

So, Why Distributed Systems? I

Centralizzato: È costoso acquistare, manutenere e aggiornare un unico, enorme e potente mainframe o server (alto costo iniziale).
Distribuito: Utilizza molti computer o nodi più economici e standard (commodity hardware), riducendo il costo per unità di potenza.

Centralizzato: Se il singolo server centrale si guasta (single point of failure), tutto il sistema va giù.
Distribuito: Se un nodo si guasta, gli altri nodi continuano a lavorare, garantendo una maggiore tolleranza ai guasti (Fault Tolerance).

Centralizzato: I dati risiedono in un unico punto, quindi sono sempre coerenti.
Distribuito: I dati sono replicati su nodi diversi. Garantire che tutte le copie rimangano aggiornate è uno dei problemi fondamentali più difficili.

Centralised vs. Distributed Systems[Puder et al., 2005]

Criteria	Centralized system	Distributed system
Economics	low	high
Availability	low	high
Complexity	low	high
Consistency	simple	difficult
Scalability	poor	good
Technology	homogenous	heterogenous
Security	high	low

Centralizzato: La crescita è limitata dalla potenza massima del singolo server. L'aggiunta di capacità è costosa e limitata.
Distribuito: La capacità può essere aumentata aggiungendo semplicemente più nodi, in modo teoricamente illimitato.

Centralizzato: Tutti i componenti hardware e software sono generalmente identici o strettamente integrati.
Distribuito: I componenti possono provenire da fornitori diversi, usare sistemi operativi e linguaggi di programmazione diversi, il che aggiunge flessibilità ma anche difficoltà di integrazione.

Centralizzato: La protezione è concentrata su un unico punto d'ingresso e confine ben definito.
Distribuito: La superficie di attacco è molto più ampia. Ci sono molti canali di comunicazione, molti nodi e molti confini da proteggere, rendendo più difficile garantire la sicurezza a livello globale.

So, Why Distributed Systems? II

In molte situazioni reali, l'ambiente computazionale o i dati stessi non possono essere fisicamente concentrati in un unico luogo.

We need distributed systems for [Ghosh, 2014]

- geographically distributed environments
- computation speedup — I sistemi distribuiti consentono di completare attività computazionalmente intensive in minor tempo, suddividendo il lavoro tra più unità di calcolo.
- resource sharing — Permettere agli utenti di accedere e utilizzare risorse che non sono sul loro computer locale, ma sono gestite da altre parti del sistema.
- fault tolerance — La capacità di un sistema di continuare a funzionare, anche se uno o più dei suoi componenti falliscono.

Artificial Systems Nowadays I

Concepire

Conceiving and constructing artificial systems...

- ... nowadays means dealing with *distributed systems*
- whose core (nucleo) is represented by (distributed) *computational systems*
- which are to be *modelled* (modellati) and *built* (e costruiti)

Poiché la distribuzione è la norma, gli ingegneri e gli scienziati informatici non possono più usare i modelli e gli strumenti semplici dei sistemi monolitici. Devono:
- Modellare (Model): Sviluppare modelli astratti e formali che catturino le sfide della distribuzione (concorrenza, guasti, latenza, mancanza di un tempo globale).
- Costruire (Build): Implementare questi modelli utilizzando architetture, protocolli e middleware specifici per sistemi distribuiti.

Artificial Systems Nowadays II

Modelling distributed (artificial) systems...

- ... involves new (theoretical) problems
- so, it requires new theoretical frameworks, models, abstractions, techniques
- mostly, computational ones

Sebbene alcuni problemi siano fisici, la maggior parte delle soluzioni risiede nell'ambito della computazione. Si tratta di inventare algoritmi e protocolli che possano funzionare correttamente in condizioni di asincronia e fallimento.

! it is one of the main objects of study of computer science

Molti dei problemi più importanti e difficili (come garantire la sicurezza, l'affidabilità e la performance su scala globale) ricadono in questo campo.

Artificial Systems Nowadays III

Building distributed (artificial) systems...

- ... involves new (practical) problems
- so, it requires new technologies, infrastructures, methods, methodologies
- mostly, computational ones
- ! it is one of the main objects of study of computer engineering

Se la Computer Science si occupa di cosa si può fare e di come ragionare sui sistemi distribuiti, l'Ingegneria Informatica si occupa di come costruirli in modo affidabile, efficiente ed economico nella realtà.

Artificial Systems Nowadays IV

As a result...

- ... we will mix up computer science and engineering issues
  - we may also say, theoretical and methodological / technological issues

all along the course

- starting from the very beginning of the course

Why Distributed Systems?

Distributed Systems

Andrea Omicini

mailto:andrea.omicini@unibo.it
Dipartimento di Informatica – Scienza e Ingegneria (DISI)
Alma Mater Studiorum – Università di Bologna

Academic Year 2025/2026

References

[Ghosh, 2014] Ghosh, S. (2014).
Distributed Systems: An Algorithmic Approach.
Chapman & Hall/CRC Computer and Information Science Series. CRC Press, 2nd edition
DOI:10.1201/b17224

[Puder et al., 2005] Puder, A., Römer, K., and Frank, P. (2005).
Distributed Systems Architecture: A Middleware Approach.
Morgan Kaufmann, 1st edition
https://shop.elsevier.com/books/distributed-systems-architecture/puder/978-0-08-045470-2