POLITECNICO
MILANO 1863

Data Lakes and Problems and future trends in (big) data Integration

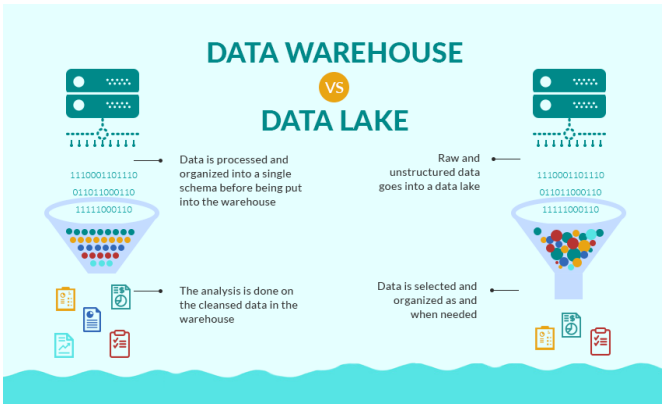
Cinzia Cappiello
A.A. 2023-2024

The Data Lake

Many organizations rely on a data warehouse, which is inadequate since it lacks the necessary flexibility because of:

- Its inherently centralized architecture
- The variability of the required data analysis
- The needs related to increasing volume and variety.

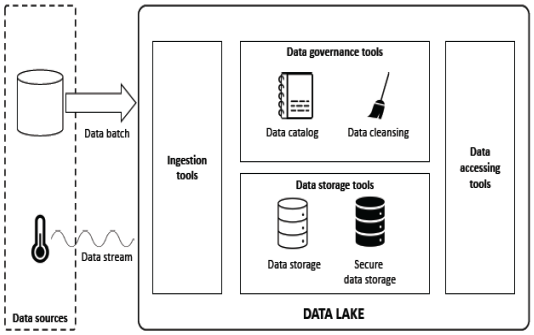
Data Lake



https://www.grazitti.com/blog/data-lake-vs-data-warehouse-which-one-should-you-go-for/

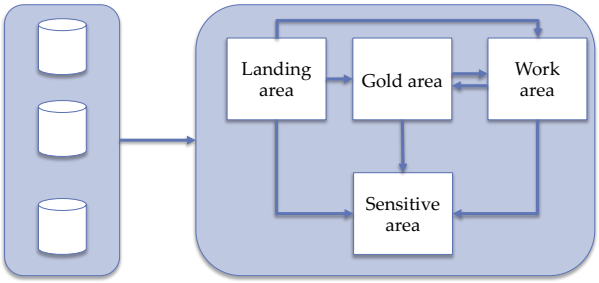
3

Data Lake Minimal Architecture



4

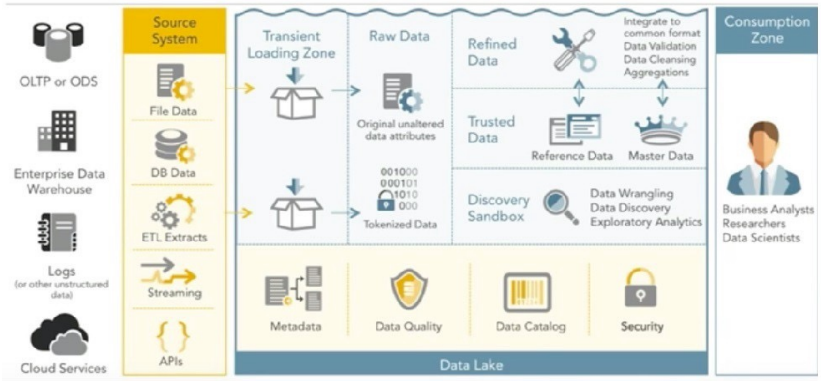
Organizing the Data Lake



Alex Gorelik, The Enterprise Big Data Lake, O'reilly 2019

5

Data lake reference architecture



https://medium.com/@rathi.ankit/data-lakes-in-modern-data-architecture-90d8c38010d

6

Data Lake design

- Data Lake can be designed on premises or in cloud
- Data Lake can be realized also as a virtual data lake (ingest on demand)

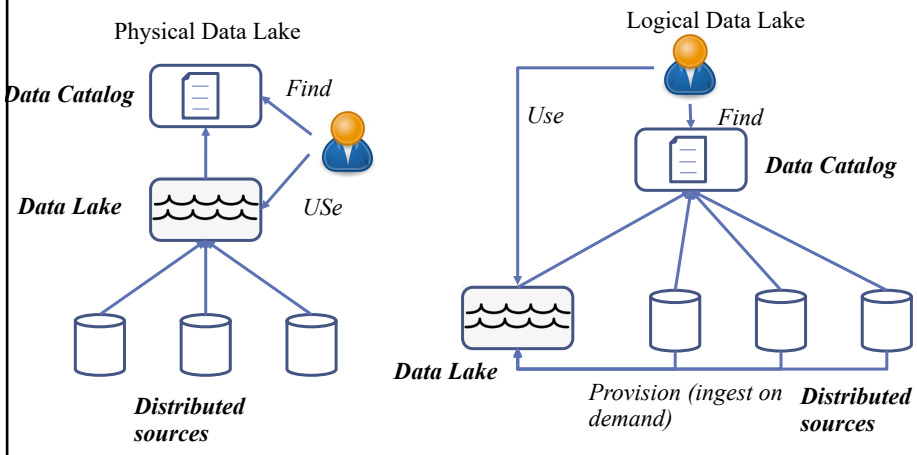
7

Main tasks of the data catalog

- Define tags for attributes and sources using consistent terms
- Provide an interface that allows users to find data sources using terms they are accustomed to
- Provide a description of the content of the sources through appropriate tags
- Include details about the provenance of the data (e.g., where it came from, whether and what preprocessing operations were performed on it)

8

Role of the Data Catalog



Alex Gorelik, The Enterprise Big Data Lake: Delivering the Promise of Big Data and Data Science, 2019

9

Metadata

- **Descriptive metadata** add information about who created a resource, what it is about and what it includes. This is best applied using semantic annotations.
- **Structural metadata** include data about the way data elements are organized, their relationships and the (possible) structure they exist in.
- **Administrative metadata** provide information about the origin of resources, their type and access rights.
- **Semantic metadata** to interpret the meaning of the data via references to concepts, often formally described in a knowledge graph or ontology.

A different, more coarse-grained classification considers:

- **Technical Metadata,** related to the profile of the datasets (e.g., list of attributes, data types, statical information), and
- **Business Metadata**, that provide a domain-aware description of a dataset.

10

Technical metadata

- **Data Profiling:** the activity that extracts metadata from data analysis. Data profiling highlights the schema, data types, formats used, maximum, minimum, average (if numeric) values, and other information such as:
  - Cardinality (number of unique values per attribute)
  - Selectivity (a measure of the uniqueness of values, measured by dividing cardinality by the number of rows)
  - Density (number of nulls per attribute)

11

Business Metadata

- Business metadata are often derived from glossaries, taxonomies and ontologies
- Business ontologies are often very complex. That is why in their place many times so-called folksonomies are used: user-built ontologies in which recurring terms are usually used and classified.

12

Tagging

- Tagging is the process by which metadata is associated with data sources.
- Tagging can be:
  - Manual
  - Crowdsourced based
  - Automatic

13

Other relevant metadata

- Metadata related to security and data protection
- Data Quality
- Data Lineage (or Provenance)

14

Data lake federation

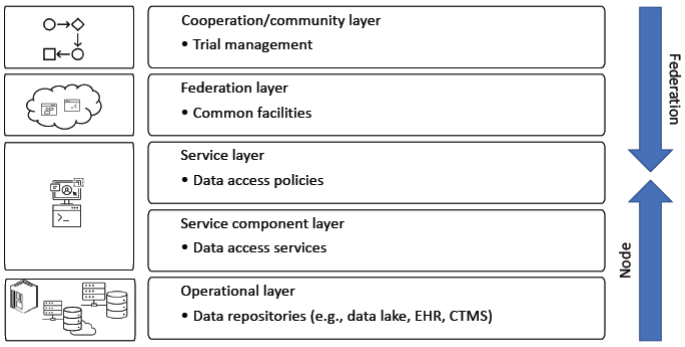
Adoption of a data lake federation through which the organizations could achieve further benefits by sharing data.

The main research challenge for guaranteeing data reliability is accurate data description, to:

- document their quality,
- facilitate data discovery,
- define security and privacy policies.

15

Data Lake Federation



16

Data Lake Federation: Node level

- Operational layer: organizations, companies, associates, ….have implemented their own data repositories to store their data. These might be deployed on the organization's premises or on cloud resources, possibly already organized in a data lake or other data organization, or using a hybrid solution.
- Service component layer: it supports the implementation of the data-access services, in charge to expose the selected datasets. For each service, depending on the nature of the data and the type of support that the node wants to offer, e.g., a simple FTP connection, a REST API.
- Service layer (middle layer): it implements the policies that regulate the access to the exposed datasets through the defined services. These policies specify who has the right to access what, and also which type of analysis can be performed on a given dataset or the transformations to be applied (e.g., anonymization) before making it available to the requester.

17

Data Lake Federation: Federation level

- Federation layer: it offers both infrastructural and application services: assuming that a set of nodes offers data services according to the structure just described, the data lake federation has the role to create an ecosystem that enables data sharing. For instance, a community cloud can be established to offer common storage and computation facilities that can be used by a node to (temporarily) store data in a convenient place to increase the performance. At the same time, a distributed monitoring system can be offered, to check if the occurring data-sharing is respecting the defined access policies.
- Cooperation/community layer: it offers the tools to define and manage multi-centric operations, i.e., where the analysed data could be stored in more than one node.

18

Pay-as-you-go approach in the data lake(s)

When a query or an update is submitted to an integration system, this has to perform some **reformulation** in terms of a set of queries over the single datasets:

- The system determines **which datasets are needed** to answer the query
- If more than one dataset is needed, determine **which predicates (conditions)** apply to only a single dataset and which predicates apply to elements from more than one dataset.
- The latter can only be evaluated over all the involved datasets, appropriately combined, whereas each of the former can be evaluated over the specific dataset addressed by that (sub) query.

In a data lake, these operations are performed in a pay-as-you-go fashion, avoiding the creation of mediated schemas or the permanent establishment of relationships between the elements of different sources (entity resolution and data fusion), and relying instead on temporary links based on metadata.

Basic tools:

- **keyword search** over a collection of data coupled with effective data visualization
- **improving the metadata** in the system to the end of supporting and validating schema and instance, instance and reference reconciliation

19

Integrating heterogeneous data into data lakes

- How does the Data Lake 'put together' all the data related to a specific query?
- How do we compare data of any kind (e.g., text, images, time series,...) to the end of deriving their similarities and differences?
  - The integration of natural language text can be supported by semantic techniques (ontologies are numerous and rich in the health domain) and/or on ML tools (e.g. word and sentence embeddings)
  - Queries that require to retrieve images, or time series, that show some resemblance, might be based on keyword-search on the image features or other metadata, or exploit some service based on image or plot similarity
  - The data lake should be able to establish a relationship between datasets that have different characteristics, and also allow different modes of interaction
  - Fundamental a catalog for describing data and services
  - This information is the basis to allow search, query and update over all the contents of the data lake, regardless of their formats

20

DW vs Data Lakes

Data Warehouse	Dataspace
Usually Relational DBMS	Often HDFS/Hadoop
Data of recognized high value	Candidate data of potential value
Aggregated data	Detailed source data
Data conforms to enterprise standards	Fidelity to original format and condition (transformed/cleaned later, on demand, when needed)
Known entities	Raw material (metadata) to discover entities
Data integration up front	Data integration on demand
Schema on write	Schema on read (when doing analytics)

Defining Data Mesh

"Decentralized Socio-Technical approach to share, access, and manage analytical data in complex and large environments – within or across organizations"

Z. Dehghani, Data Mesh. Delivering Data-Driven Value at Scale, O'Reilly, 2022

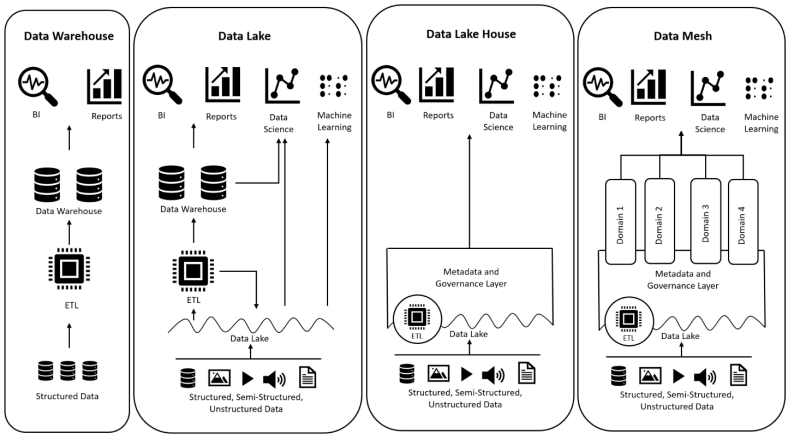
The principles of Data Mesh (1/2)

- Domain ownership
  - ownership shifts from the centralized ETL pipelines to decentralized units closer to the data following the principles of Domain-Driven Design
- Data as a product
  - data teams must apply product thinking to the datasets provided
  - ensuring discoverability and quality-control.
  - DATSIS principles: Discoverability, Addressability, Trustworthiness, Self-describing, Interoperability and Security

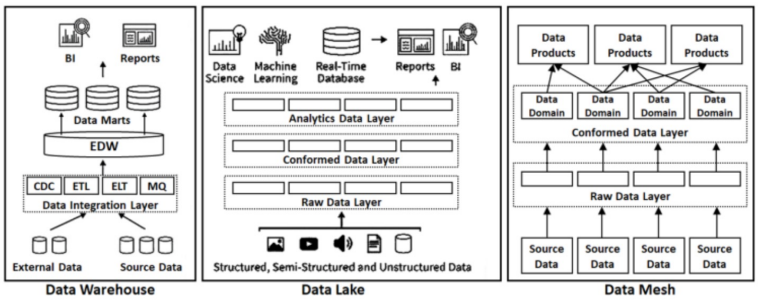
The principles of Data Mesh (2/2)

- Self-serve data platform
  - Data product lifecycle should be supported by a self-serve data platform, providing the needed tools and infrastructures and accessible without specialized knowledge
- Federated Computational Governance
  - Ensuring data ownership, standardization, interoperability and automated decision execution

Data Mesh



https://www.linkedin.com/pulse/data-warehouse-vs-lake-lakehouse-mesh-ashish-bijawat/



https://www.linkedin.com/pulse/data-warehouse-mesh-making-right-choice-chandra-kallur/

Comparison

Data Design Patterns	Data Warehouse	Data Lake	Data Lakehouse	Data Mesh
What it is	A subject-oriented data architecture that integrates detailed data in a consistent way while maintaining a non-volatile history of it.	A set of long-term data containers for managing and refining raw data, using low-cost object storage often delivered from the cloud.	A combination of a data warehouse and a data lake.	A domain-driven data design pattern divided either logically or physically among the teams working in those domains.
Benefits	Generates actionable insights (e.g., in dashboards) from huge amounts of curated data, including the creation of predictive analytics and dashboards that drive operational actions. It aggregates data from all enterprise sources in a central location with consistent governance and supports sandboxes for new idea testing.	Captures previously discarded "dark data" to drive innovation later on and stores data as-is without having to structure it first. The lake also allows insights to be efficiently captured by AI and machine learning services analysing raw information.	Enables an enterprise to systematically extract insights in the mode of a data warehouse — via SQL, machine learning, or any other process — while taking advantage of the vast scale and low costs of a data lake.	Data mesh allows for autonomous active management of data by the teams closest to it and permits increased agility because there's no central bottleneck. Each team can create its own data products.
Limitations	Not ideal for big data use cases that require the storage and extraction of value from large amounts of raw data, such as that created by IoT devices and web and mobile sources.	Relatively few off-the-shelf tools are available for data lakes, which necessitates significant experience with open-source software. There's also a high risk of silos due to limited governance, and there can be great difficulty balancing issues between security and ease of access.	Limited agility in adding new features because everything is centralized and monolithic. Data engineers end up spending a lot of their time cleaning up data from teams that have limited incentive to ensure their information is accurate as it goes in.	It's a relatively new architecture that enterprises are still working out. Performance and governance may suffer because users need to go over the network every time to access different data. Without cross-domain governance and semantic linking of data, it can become very siloed and yield disappointing results.

https://www.linkedin.com/pulse/data-warehouse-vs-lake-lakehouse-mesh-ashish-bijawat/

Big data integration

Data Integration: architecture and three major steps

Luna Dong and Srivastava 2015

29

Uncertain Data

- Databases are assumed to represent *certain data*: a tuple in the database is true, any tuple NOT in the database is false (Closed World Assumption)
- Real life is not as certain! Examples:
  - Observations (e.g. sensor readings) may be *unreliable*
  - A person might be still *undecided* as to a certain choice
- *Uncertain databases* of various kinds (e.g. based on probability) attempt to model uncertain data and to answer queries in an uncertain world

30

Uncertainty in Data Integration

- Data itself may be uncertain (e.g. extracted from an unreliable source)
- Mappings might be approximate (e.g. created by relying on automatic ontology matching)
- Reconciliation is approximate
- Approximate mediated schema
- Imprecise queries, such as keyword-search queries, are approximate

31

Schema alignment

- There can be thousands to million of data sources in the same domain but they often describe the domain using different schema

Luna Dong and Srivastava 2015

32

Probabilistic schema alignment

- Uncertainty on how to model the domain
- Uncertainty on the creation of the mediated schema
- Two solutions:
  - Probabilistic mediated schema
  - Probabilistic schema mapping

Luna Dong and Srivastava 2015

Probabilistic mediated schema

- The mediated schema is the set of schema terms (e.g., relational table, attribute names) on which queries are posed.
- A probabilistic mediated schema (p-med-schema) consists of a set of mediated schemas, each with a probability indicating the likelihood that the schema correctly describes the domain of the sources.

S1(Flight Number (FN), Departure Gate Time (DGT), Takeoff Time (TT), Landing Time (LT), Arrival Gate Time (AGT))

S2(Flight Number (FN), Departure Time (DT), Arrival Time (AT))

Med1({FN}, {DT, DGT, TT}, {AT, AGT, LT})
Med2({FN}, {DT, DGT}, {TT}, {AT, LT}, {AGT})
Med3({FN}, {DT, DGT}, {TT}, {AT, AGT}, {LT})
Med4({FN}, {DT, TT}, {DGT}, {AT, LT}, {AGT})
Med5({FN}, {DT}, {DGT}, {TT}, {AT}, {AGT}, {LT})

Possible Mediated Schema	Probability
Med3({FN}, {DT, DGT}, {TT}, {AT, AGT}, {LT})	0.6
Med4({FN}, {DT, TT}, {DGT}, {AT, LT}, {AGT})	0.4

Luna Dong and Srivastava 2015

Probabilistic mediated schema

- Das Sarma et al. [2008] propose an algorithm for creating a probabilistic mediated schema for source schemas S1, . . . , Sn: first construct the multiple mediated schemas Med1, . . . , Medl in pMed, and then assign each of them a probability.
- Two pieces of information available in the source schemas can serve as evidence for attribute clustering:
  - pairwise similarity of source attributes: indicates when two attributes are likely to be similar, and is used for creating multiple mediated schemas
  - statistical co-occurrence properties of source attributes: indicates when two attributes are likely to be different, and is used for assigning probabilities to each of the mediated schemas

Luna Dong and Srivastava 2015

35

Probabilistic schema mapping

- Schema mappings describe the relationship between the contents of the sources and that of the mediated data. In many applications it is impossible to provide all schema mappings upfront. Probabilistic schema mappings can capture uncertainty on mappings between schemas.

Possible Mapping Between S1 and Med3		Probability	Possible Mapping Between S1 and Med4		Probability
M_1	{(FN, FN), (DGT, DDGT), (TT, TT), (AGT, AAGT), (LT, LT)}	0.64	M_5	{(FN, FN), (DGT, DGT), (TT, DTT), (AGT, AGT), (LT, ALT)}	0.64
M_2	{(FN, FN), (DGT, DDGT), (TT, TT), (AGT, LT), (LT, AAGT)}	0.16	M_6	{(FN, FN), (DGT, DGT), (TT, DTT), (AGT, ALT), (LT, AGT)}	0.16
M_3	{(FN, FN), (DGT, TT), (TT, DDGT), (AGT, AAGT), (LT, LT)}	0.16	M_7	{(FN, FN), (DGT, DTT), (TT, DGT), (AGT, AGT), (LT, ALT)}	0.16
M_4	{(FN, FN), (DGT, TT), (TT, DDGT), (AGT, LT), (LT, AAGT)}	0.04	M_8	{(FN, FN), (DGT, DTT), (TT, DGT), (AGT, ALT), (LT, AGT)}	0.04

Similarly, there is a probabilistic mapping between S1 and Med4, where DTT ={ DT, TT } and ALT ={ AT, LT }.

Luna Dong and Srivastava 2015

36

Data Provenance

37

Data Provenance

- Also called data lineage or data pedigree. Sometimes knowing where the data have come from and how they were produced is critical.
- Provenance of a data item records "where it comes from":
  - Who created it
  - When it was created
  - How it was created - as a value in a database, as the result of a computation, coming from a sensor, etc...
- E.g. an information extractor might be unreliable, or one data source is more authoritative than others
- The database community models provenance in terms of how the datum was derived from the original source databases, the rest is left to the application (it is assumed to be domain dependent)

38

Two Viewpoints on Provenance

- *Provenance as Annotations on Data:* models provenance as *a series of annotations describing how each data item was produced*. These annotations can be associated with tuples or values
- *Provenance as a Graph of Data Relationships: it* models provenance *as a (hyper)graph, with tuples as vertices*. Each possible direct derivation of a tuple from a set of source tuples is a hyper-edge connecting the source and derived tuples

The two views are equivalent, and it is possible to convert one into the other as convenient

39

Uses of provenance information

- Explanations
- Scoring of sources and data quality
- Influence of sources on one another

40

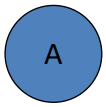
A model of provenance: Open Provenance Model (OPM)

- Allows us to express all the causes of an item
- Allow for process-oriented and dataflow oriented views
- Based on a notion of annotated causality graph

41

Nodes

- **Artifact**: Immutable piece of state, which may have a physical embodiment in a physical object, or a digital representation in a computer system.



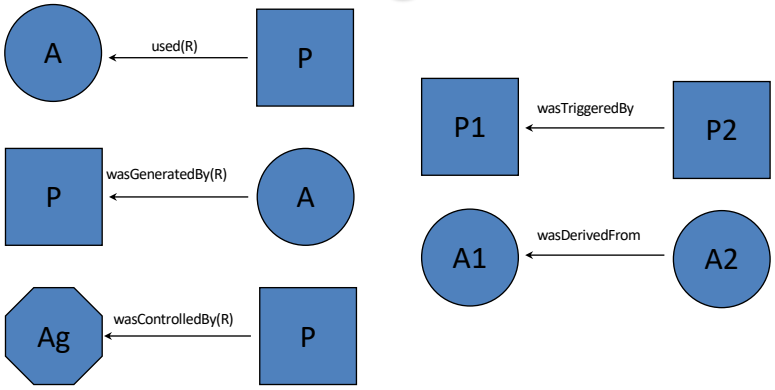
- **Process**: Action or series of actions performed on or caused by artifacts, and resulting in new artifacts.

- **Agent**: Contextual entity acting as a catalyst of a process, enabling, facilitating, controlling, affecting its execution.

[Moreau, L., *et al*. 2011]

42

Edges



[Moreau, L., *et al*. 2011]

Edge labels are in the past to express that these are used to describe past executions

43

Mashups

44

44

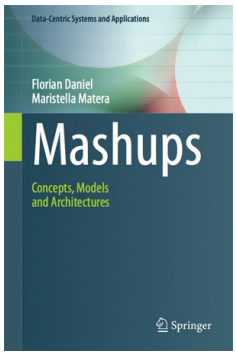
Lightweight Integration

Many data integration tasks are transient:

- We may need to integrate data from multiple sources to answer a question asked once or twice. The integration needs to be done quickly and by people without technical expertise (e.g. a disaster response situation in which reports are coming from multiple data sources in the field, and the goal is to corroborate them and quickly share them with the affected public)
- Problems typical of lightweight data integration:
  - locating relevant data sources
  - assessing source quality
  - helping the user understand the semantics
  - supporting the process of integration.
- Ideally, machine learning and other techniques can be used to amplify the effects of human input, through semi-supervised learning, where small amounts of human data classification, plus large amounts of additional raw ("unlabeled") data, are used to train the system.

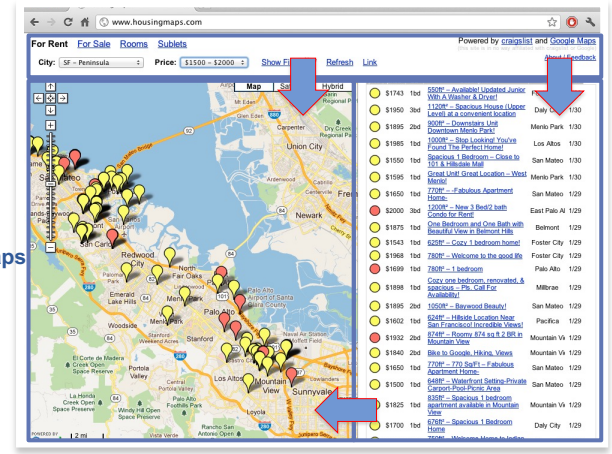
Mashups: a paradigm for lightweight integration

(from a tutorial by:
Cinzia Cappiello, Florian Daniel, Maristella Matera)



The term mashup is widely used today:

A mashup is *an application that integrates two or more mashup components at any of the application layers* (data, application logic, presentation layer) possibly putting them into communication with each other



Own application logic/UI

GoogleMaps

Craiglist

The housingmaps.com mashup

Provides for the synchronized exploration of housing offers from *craigslist.com* and maps by *Google Maps*

Integration is the added value provided by the mashup

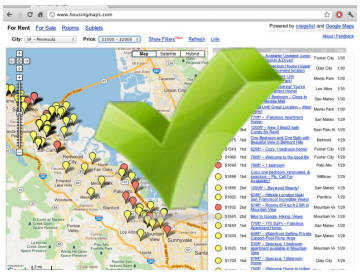
47

Key elements

Mashup component is any piece of data, application logic and/or user interface that can be reused and that is accessible either locally or remotely (e.g., Craigslist and Gmaps)

Mashup logic is the internal logic of operation of a mashup; it specifies the invocation of components, the control flow, the data flow, the data transformations, and the UI of the mashup

No added value



Additional information, functions, visualizations!

48

Types of mashup

- Data mashups
  - Fetch data from different resources, process them, and return an integrated result set
- Logic mashups
  - Integrate functionality published by logic or data components
- User Interface (UI) mashups
  - Combine the component's native UIs into an integrated UI; the components' UIs are possibly synchronized with each other
- Hybrid mashups
  - Span multiple layers of the application stack, bringing together different types of components inside one and a same application;

→ integration happens at more than one layer

49

Data mashup vs. data integration…

- Data mashups are a Web-based, lighweight form of data integration, meant to solve different problems
- Covering the "long tail" of data integration requirements
  - Very specific reports or ad-hoc data analyses
  - Simple, ad-hoc data integrations providing "situational data" that meet short term needs
  - Non-mission-critical integration requests

50

Other definitions of MashUps

"**Web-based resources** consisting of dynamic networks of interacting components" (Abiteboul et Al., 2008)

"**API enablers**" (Ogrinz, 2009), to create an own API where there is none

"**Combination of content** from more than one source into an integrated experience" (Yee, 2008)