



POLITECNICO
DI TORINO



e-Lite



«Integration» - a necessary evil

Sistemi Informativi Aziendali – A.A. 2011/2012

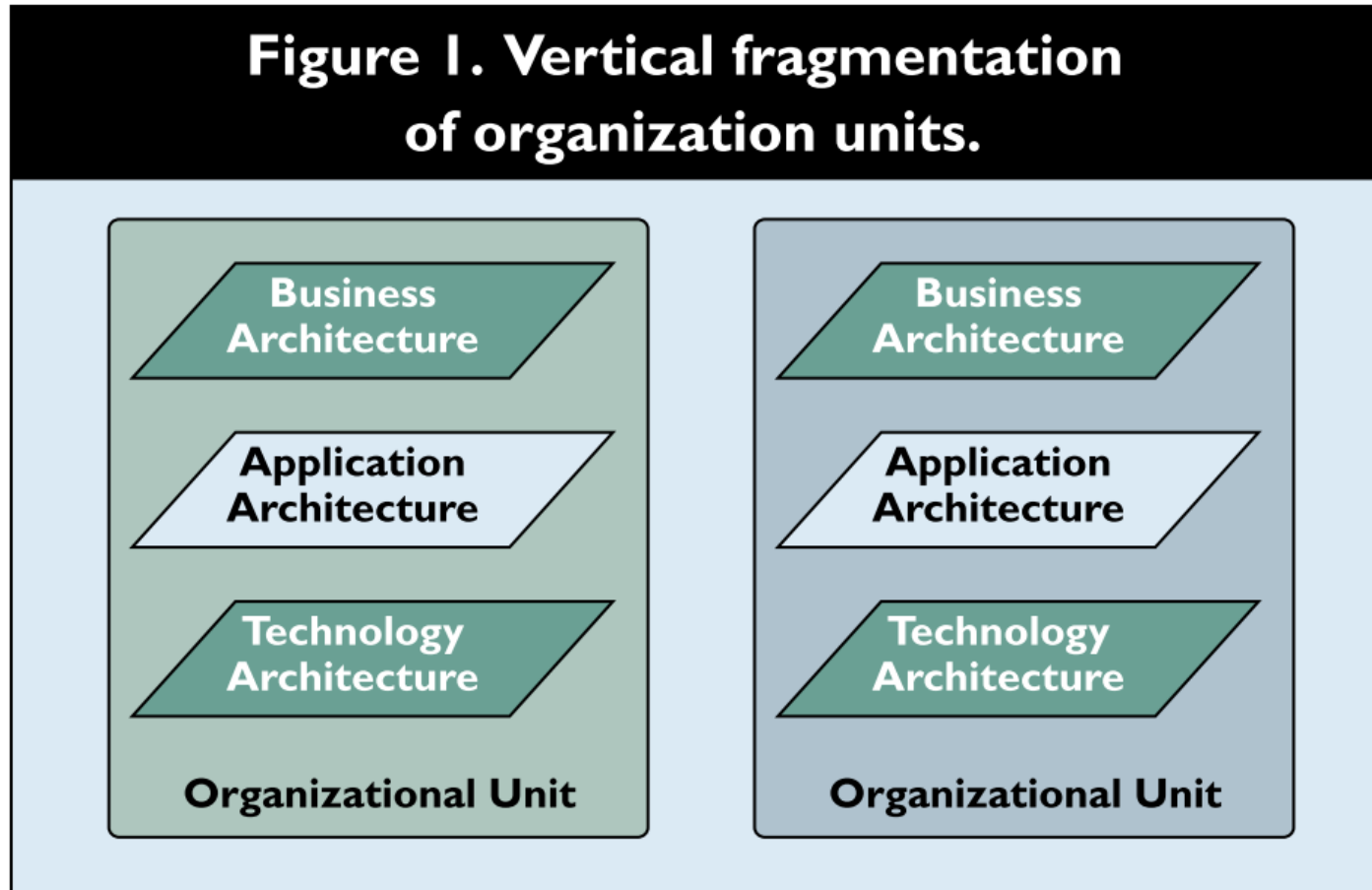
Definition

- ▶ **System (or Systems) Integration**

- ▶ In information technology, **systems integration** is the process of linking together different computing systems and software applications physically or functionally, to act as a coordinated whole.

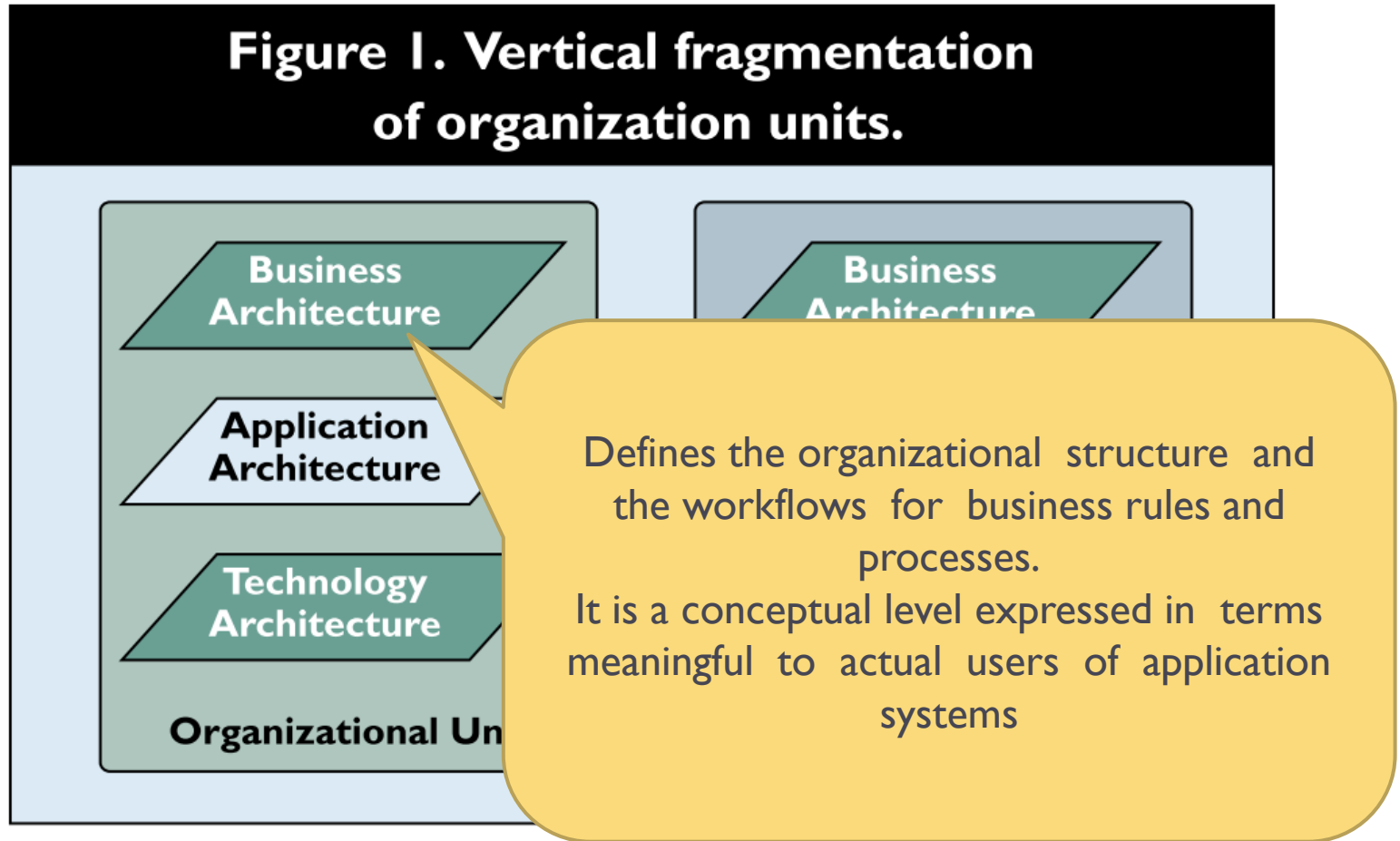
http://en.wikipedia.org/wiki/System_integration

The problem



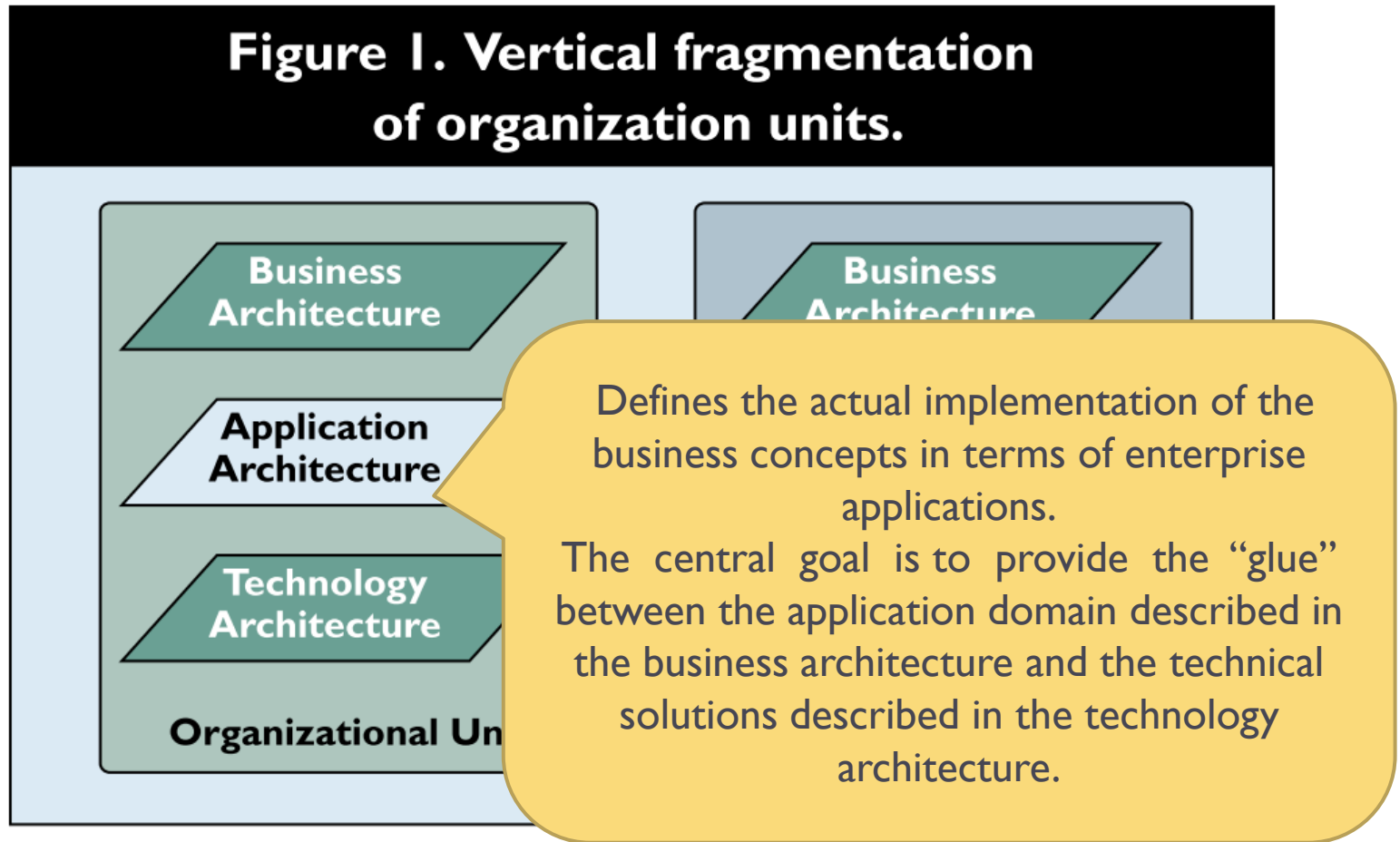
Wilhelm Hasselbring et al. (2000), "Information system integration", Communications of the ACM, Volume 43, Issue 6 (June 2000), Pages: 32-38

The problem



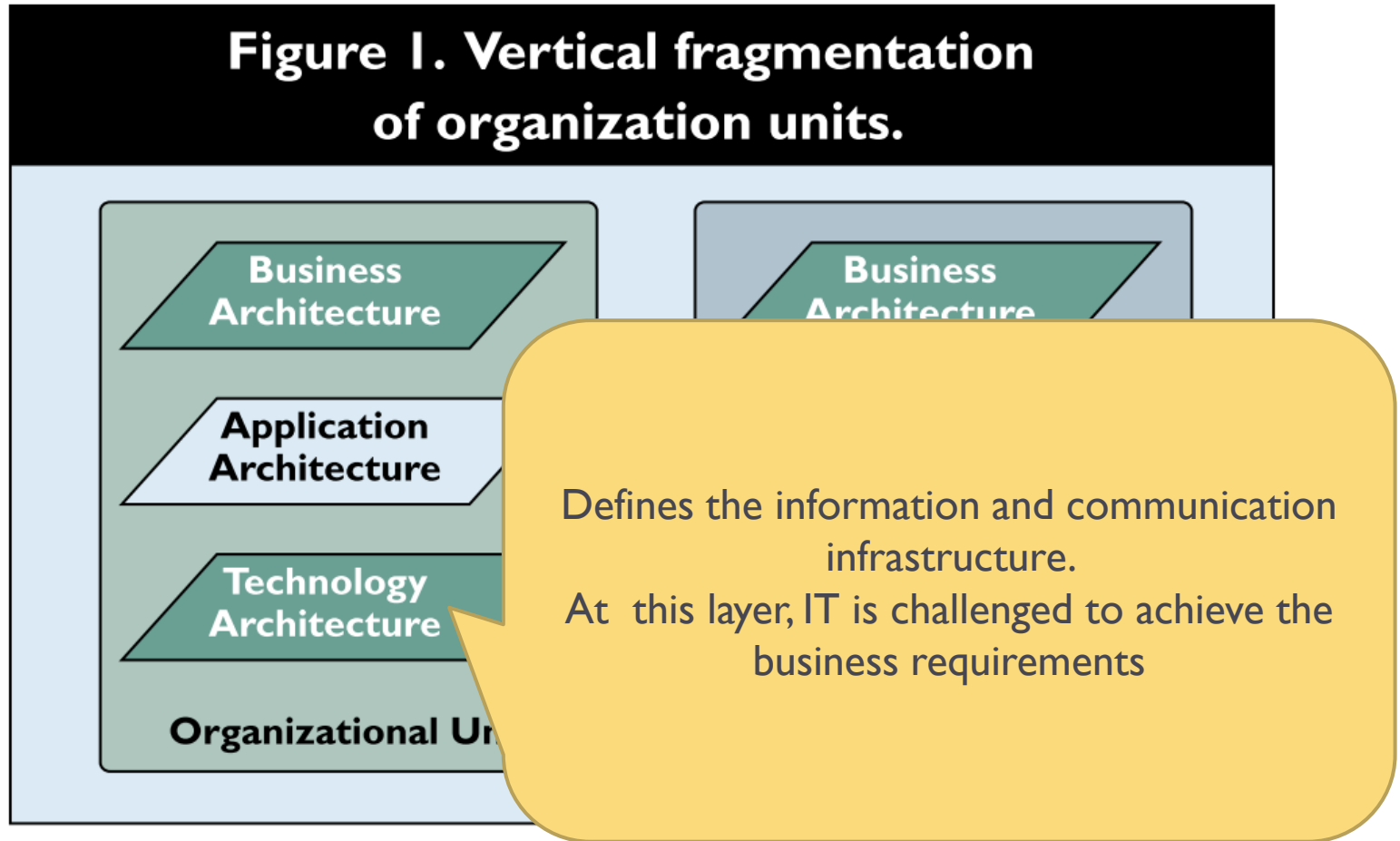
Wilhelm Hasselbring et al. (2000), "Information system integration", Communications of the ACM, Volume 43, Issue 6 (June 2000), Pages: 32-38

The problem



Wilhelm Hasselbring et al. (2000), "Information system integration", Communications of the ACM, Volume 43, Issue 6 (June 2000), Pages: 32-38

The problem



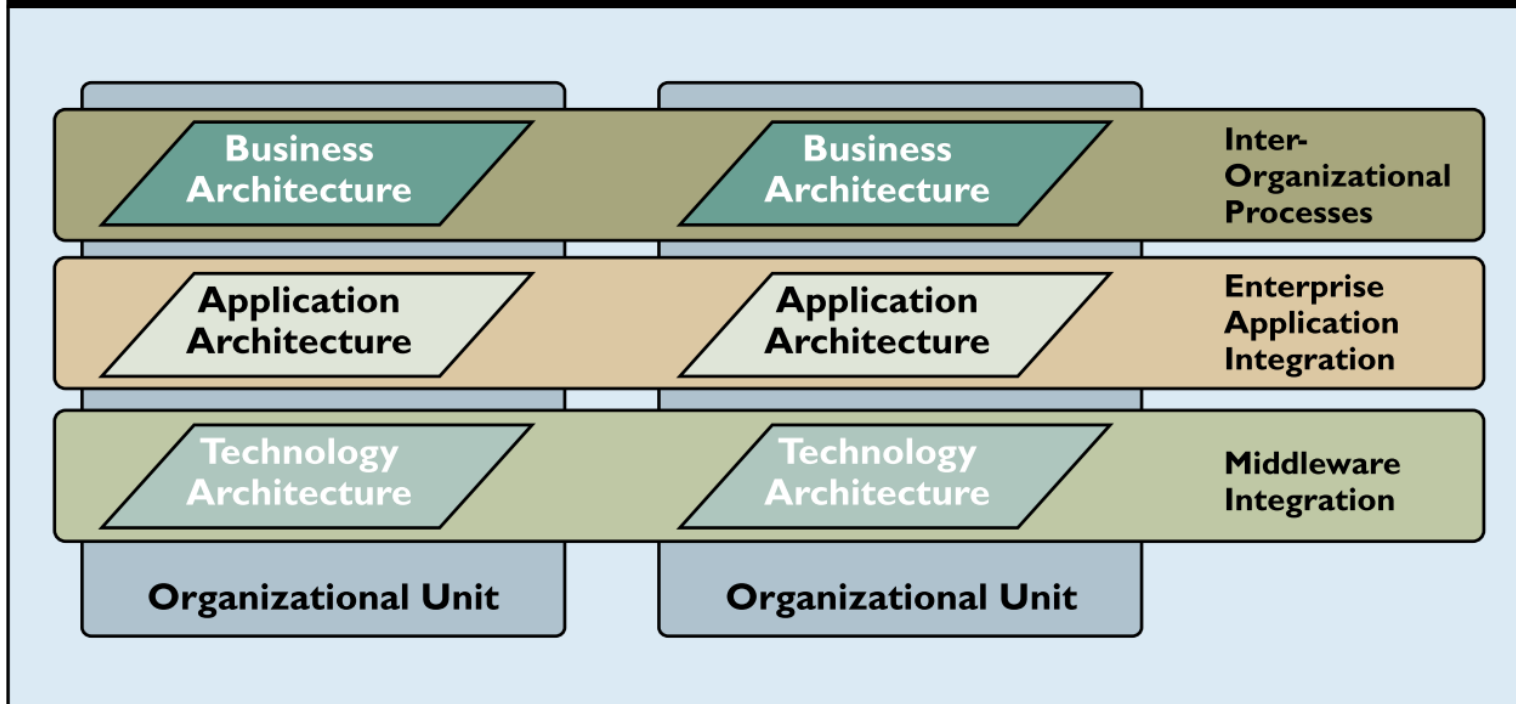
Wilhelm Hasselbring et al. (2000), "Information system integration", Communications of the ACM, Volume 43, Issue 6 (June 2000), Pages: 32-38

No Silos!

- ▶ Organizational units need to cooperate
- ▶ Cooperation at all levels is needed
- ▶ Cooperation is not “automatic” whenever different approaches/standard/methods/information is used

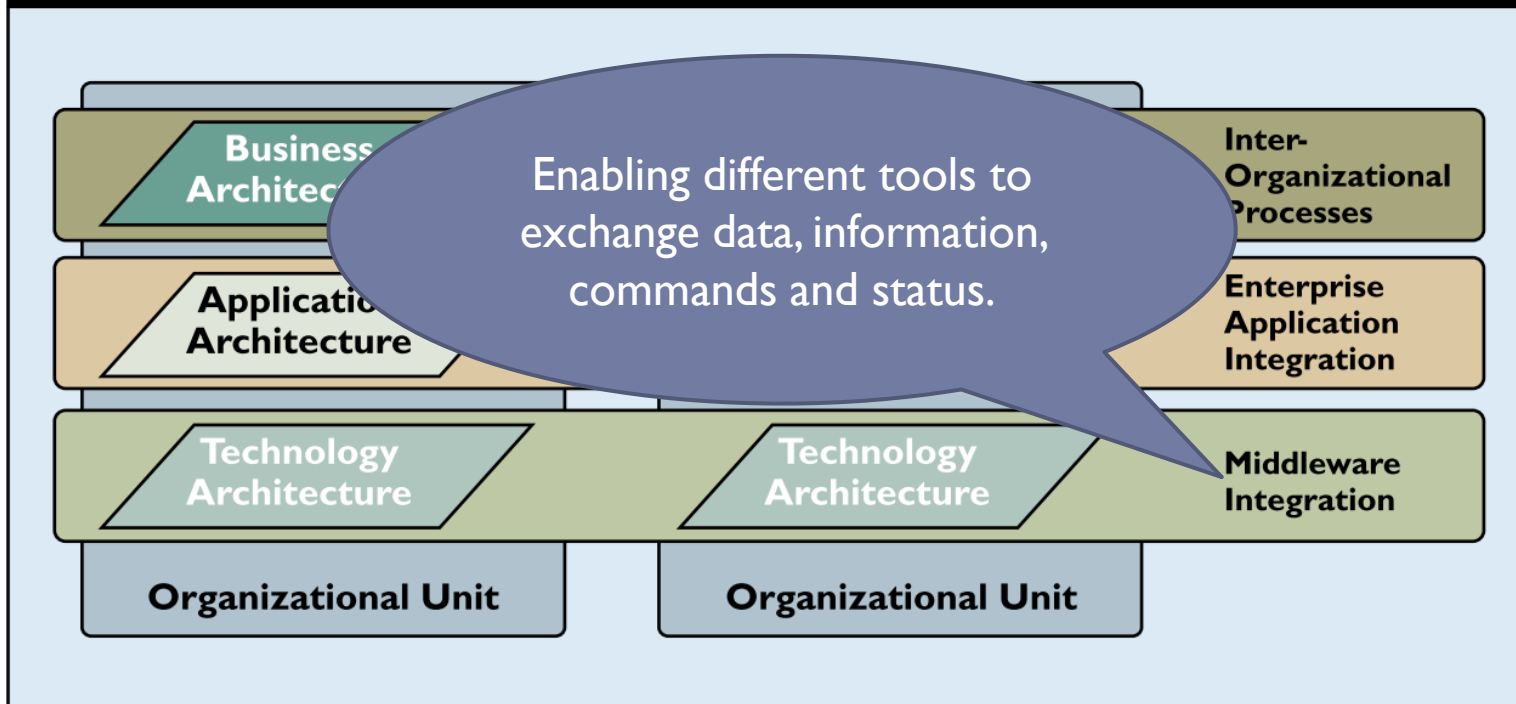
Integration levels

Figure 2. Horizontal integration to support the business processes.



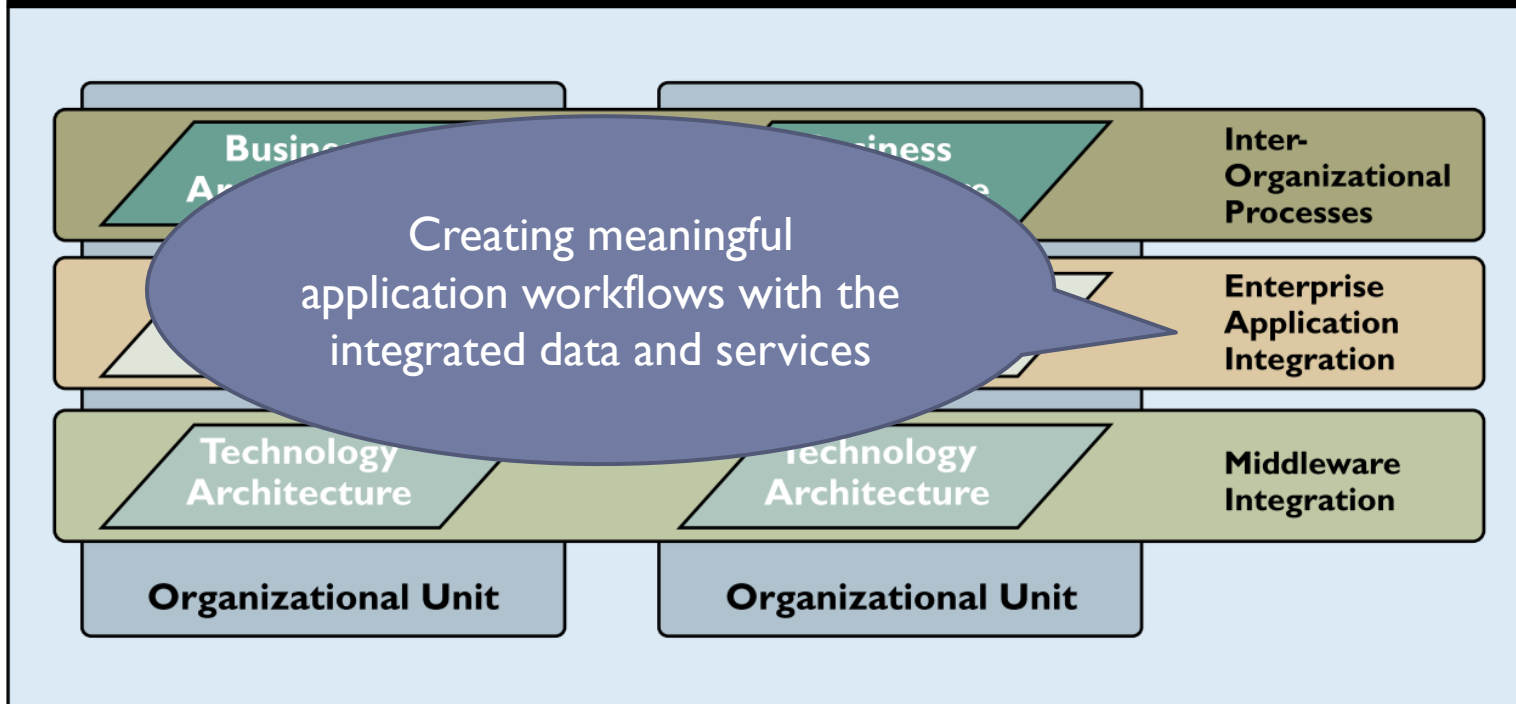
Integration levels

Figure 2. Horizontal integration to support the business processes.



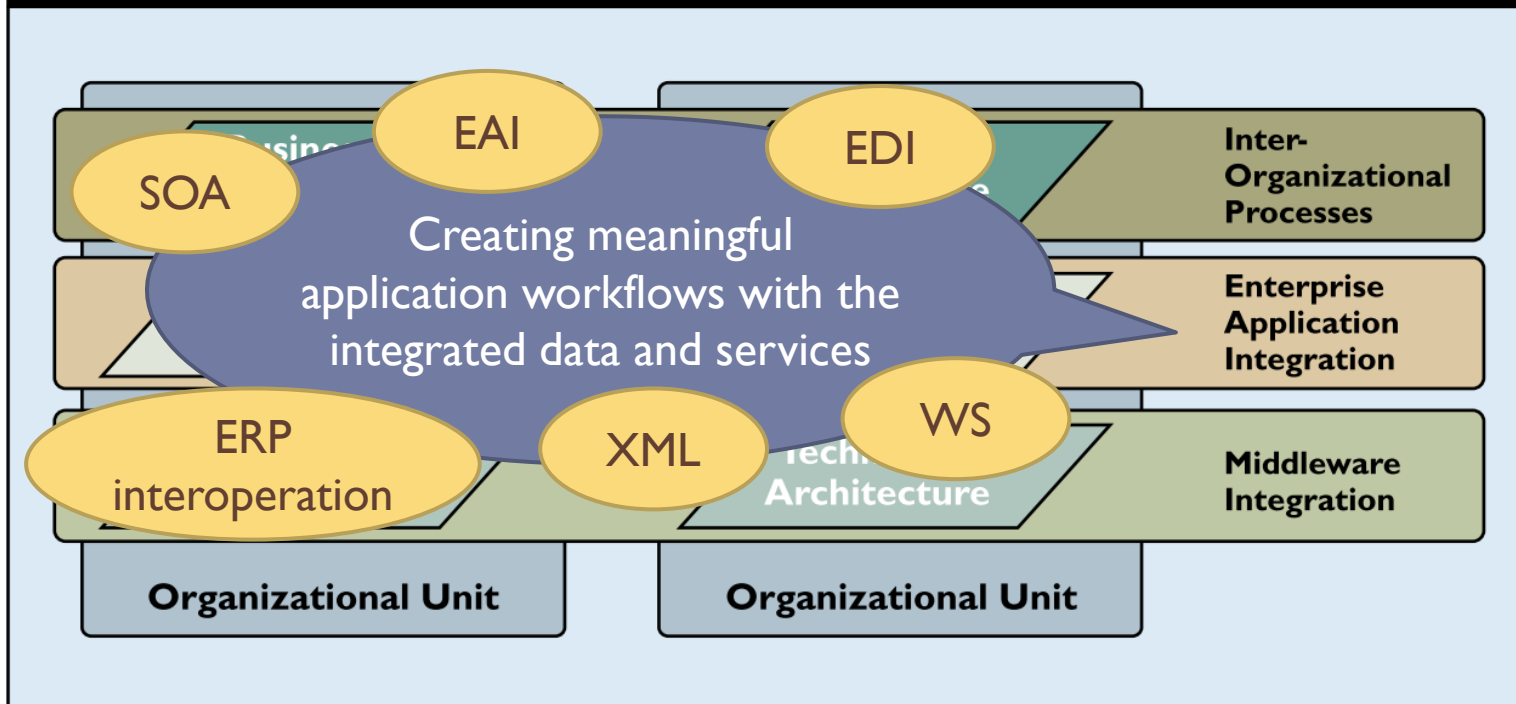
Integration levels

Figure 2. Horizontal integration to support the business processes.



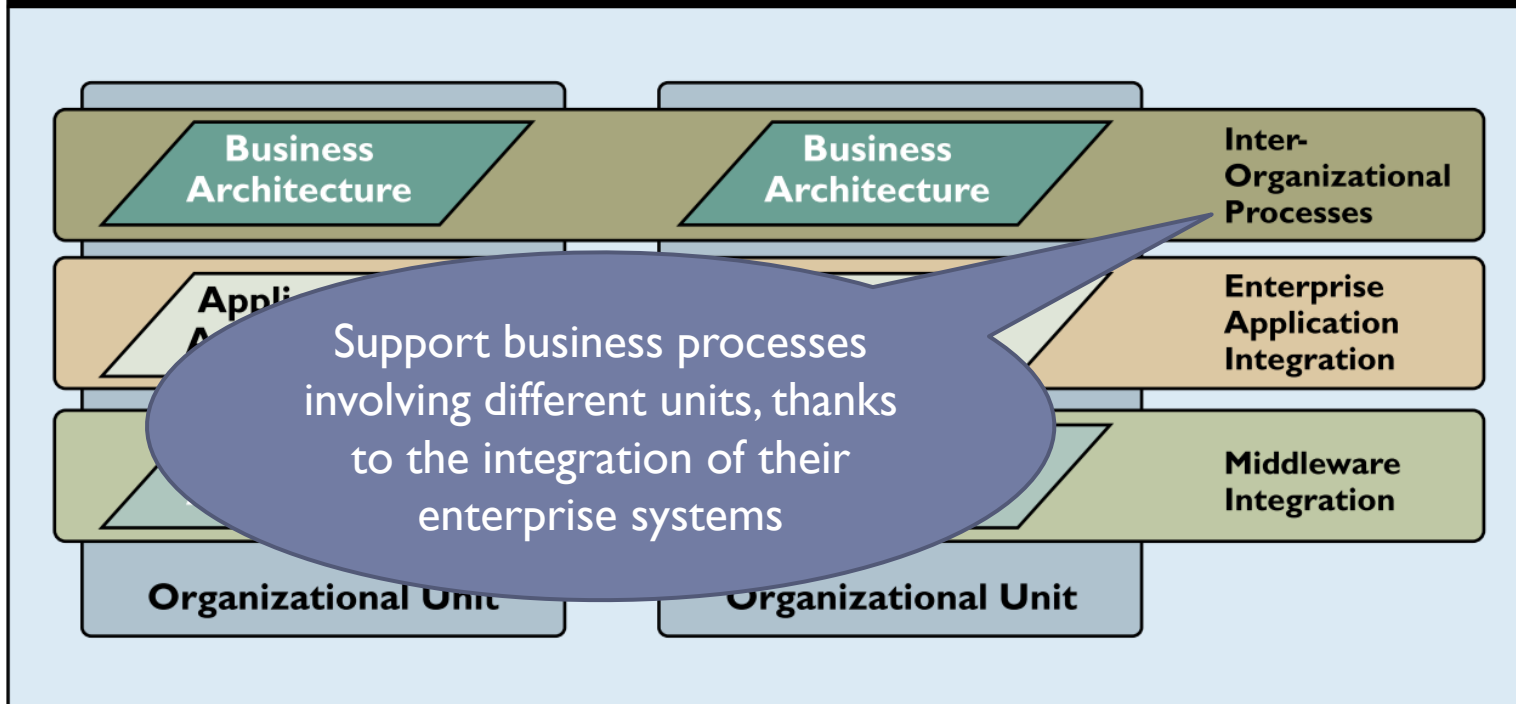
Integration levels

Figure 2. Horizontal integration to support the business processes.



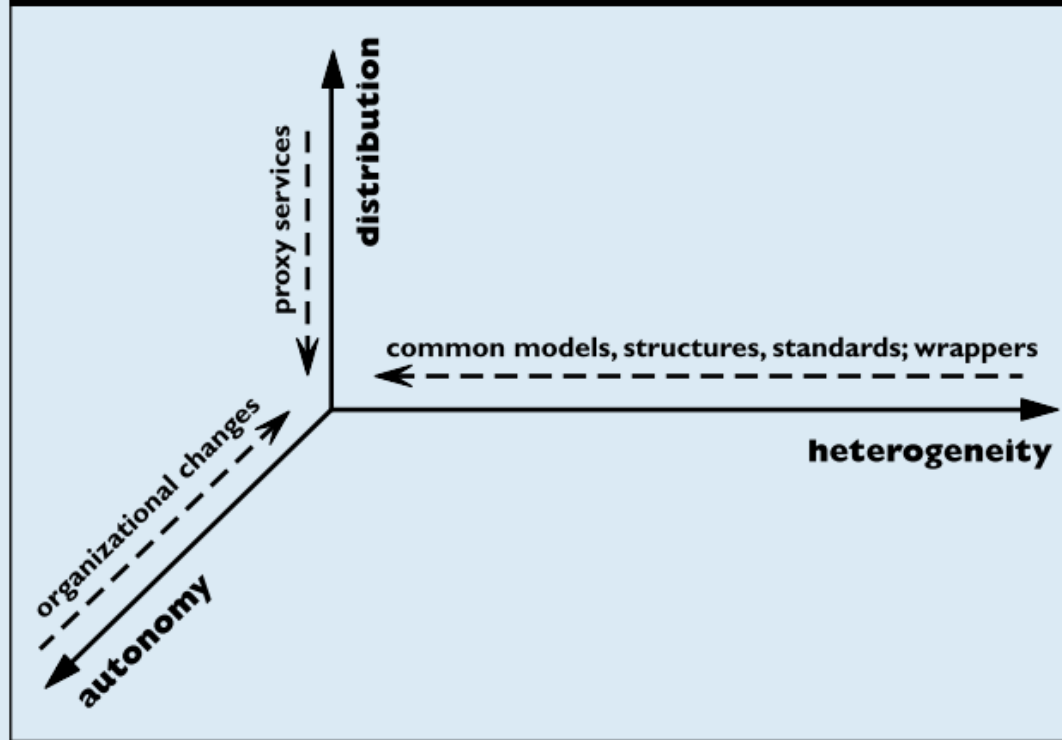
Integration levels

Figure 2. Horizontal integration to support the business processes.

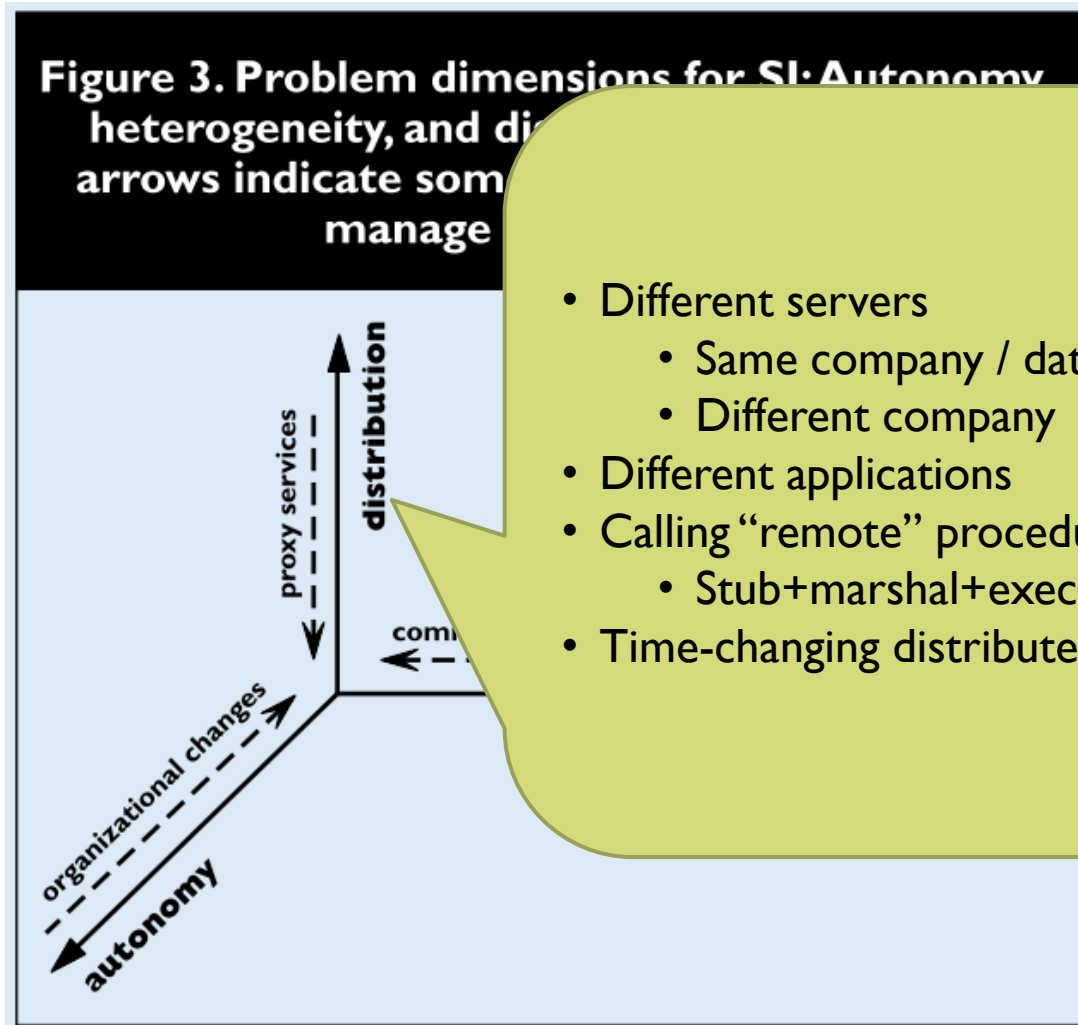


Complexity Boosters

Figure 3. Problem dimensions for SI: Autonomy, heterogeneity, and distribution. The dashed arrows indicate some general approaches to manage these issues.

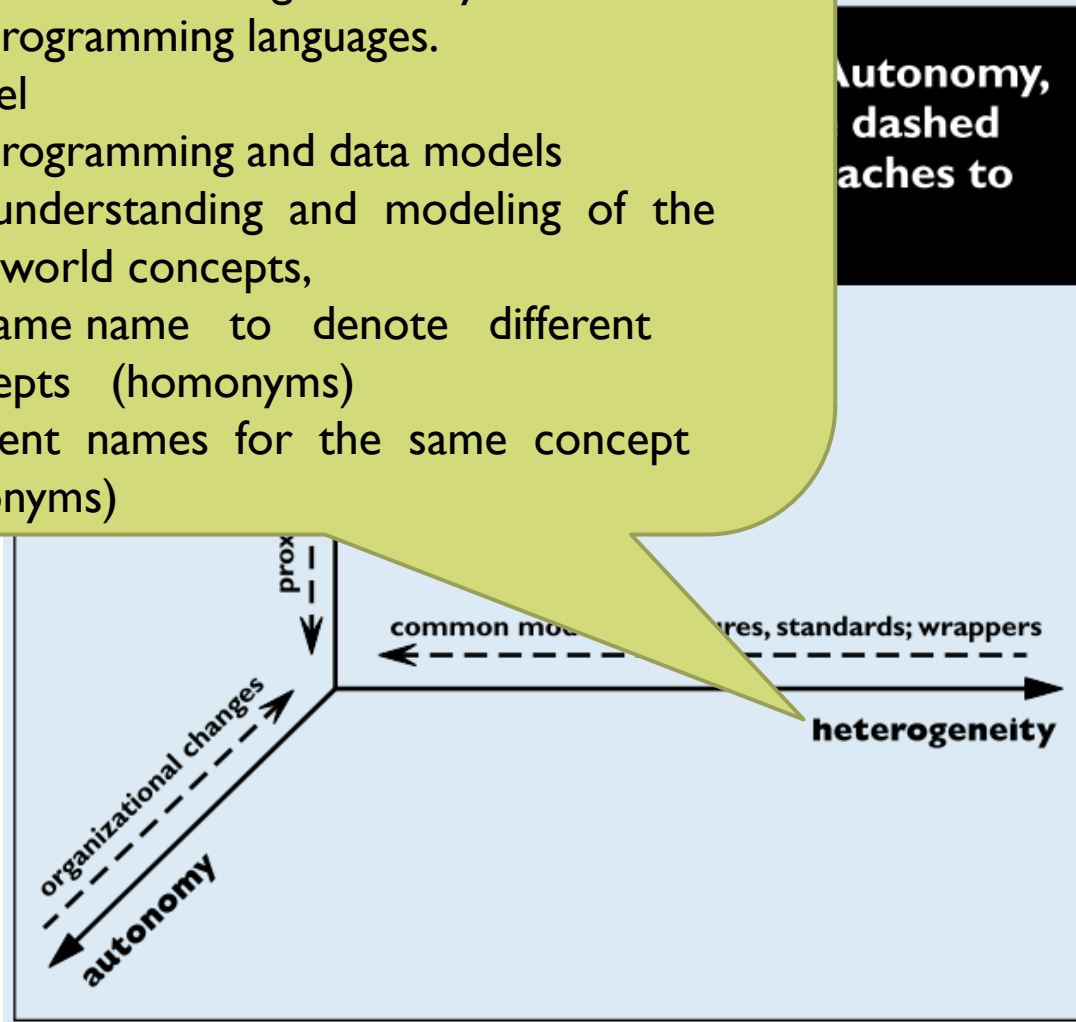


Complexity Boosters

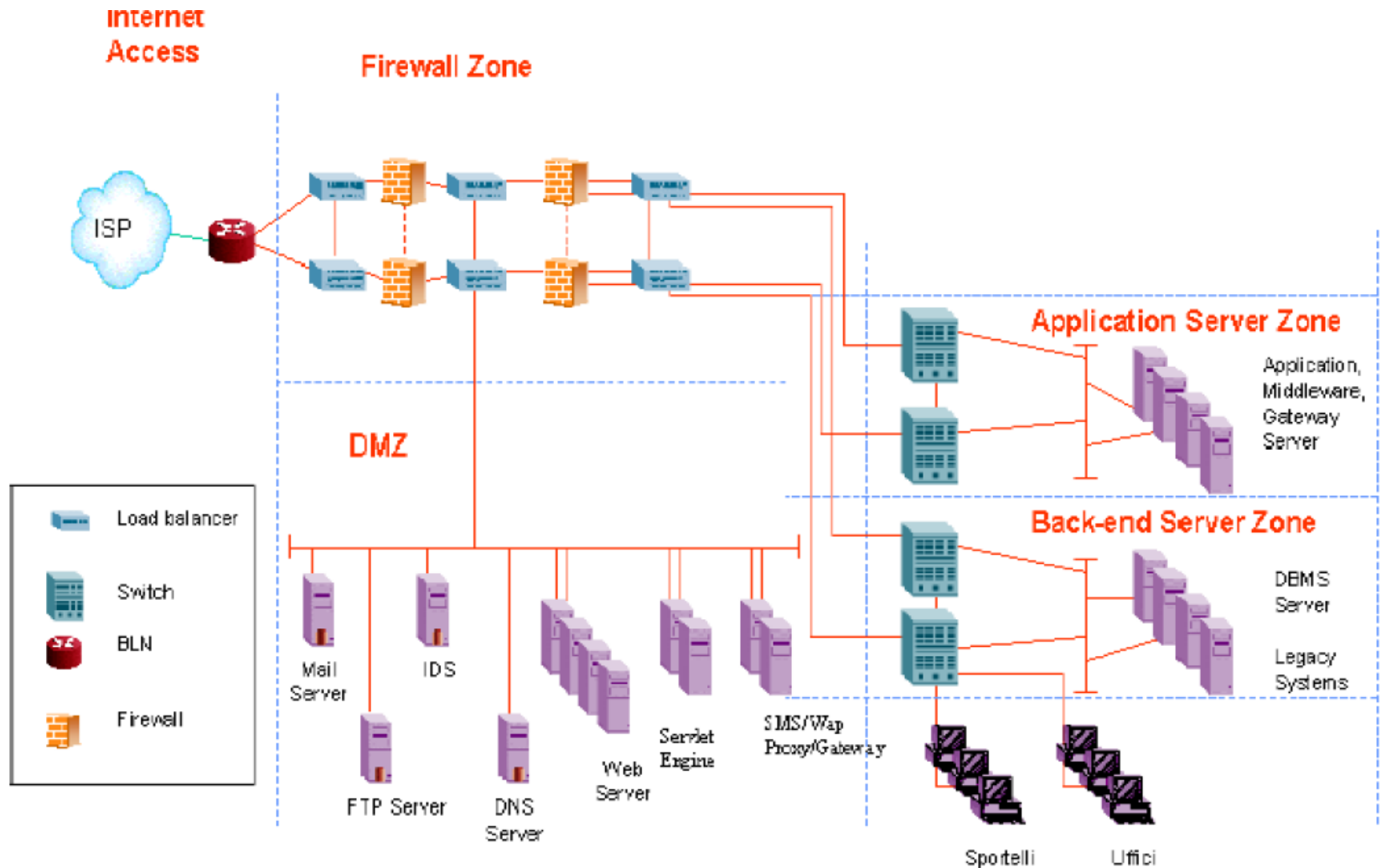


- Different servers
 - Same company / data center
 - Different company
- Different applications
- Calling “remote” procedures
 - Stub+marshal+execute
- Time-changing distributed architecture

- Technical level:
 - different hardware platforms,
 - different operating systems,
 - different database management systems
 - different programming languages.
- Conceptual level
 - different programming and data models
 - different understanding and modeling of the same real-world concepts,
 - the same name to denote different concepts (homonyms)
 - different names for the same concept (synonyms)



The “cutting game”



Integration styles

▶ File Transfer

- ▶ Have each application produce files of shared data for others to consume, and consume files that others have produced

▶ Shared database

- ▶ Have the applications store the data they wish to share in a common database

▶ Remote Procedure Invocation

- ▶ Have each application expose some of its procedures so that they can be invoked remotely, and have applications invoke those to run behavior and exchange data

▶ Messaging

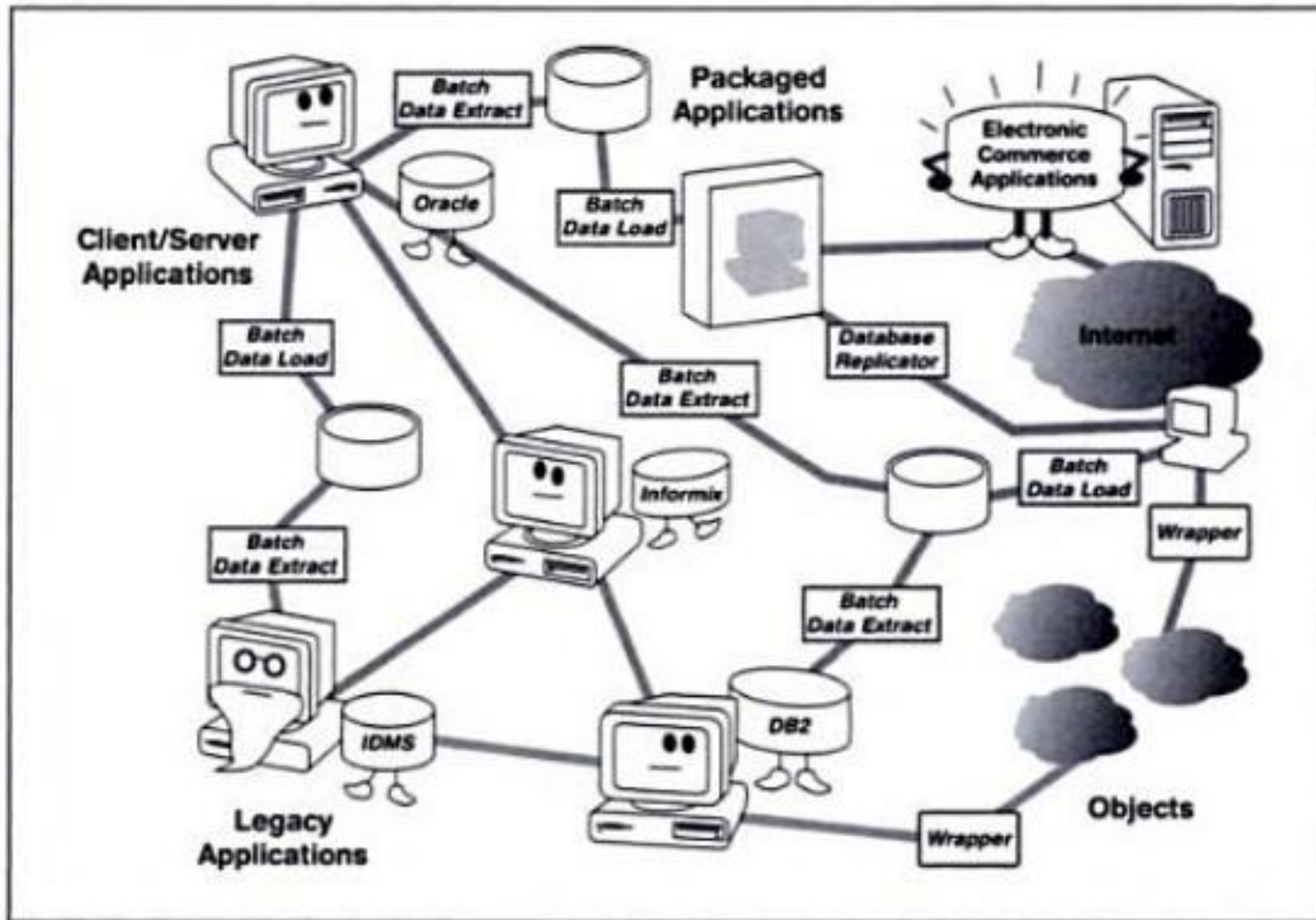
- ▶ Have each application connect to a common messaging system, and exchange data and invoke behavior using messages

Available Technical solutions

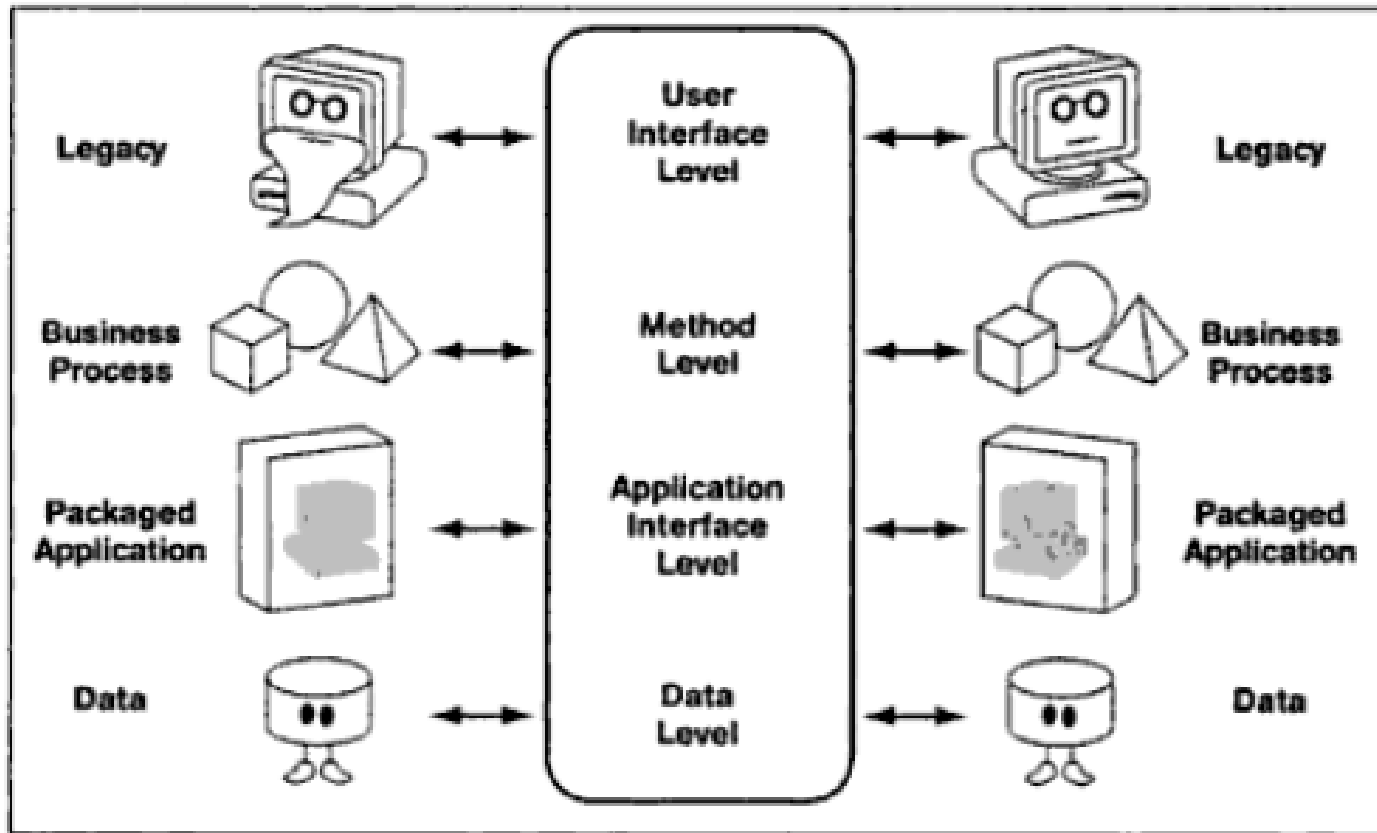
- ▶ Messaging oriented middleware; messaging patterns
- ▶ Interface based middleware; distributed objects
- ▶ Database-Oriented Middleware
- ▶ XML; XML based languages and standards
- ▶ Service Oriented Architecture; Web Services
- ▶ Web 2.0: Web APIs, REST, Mashup
- ▶ REST; RESTful Web Services; REST vs. Web Services
- ▶ Data/information integration and EII
- ▶ Presentation integration and portal
- ▶ Process (oriented) integration
- ▶ Identity integration; Single Sign-On

CIS 8020: Systems Integration, Georgia State University
<http://www2.cis.gsu.edu/cis/program/syllabus/graduate/cis8020.asp>

Legacy systems can't be killed



Legacy systems integration



<http://www.ibm.com/developerworks/webservices/library/ws-tip-leg/index.html>

Data-level integration

- ▶ Access of **legacy databases** or **files** by either session beans or entity beans
- ▶ Access to **newly developed data feeds**, produced by the legacy system, for the specific purpose of data access
- ▶ **XML** is often used
- ▶ Straightforward and quick to implement
- ▶ **But:**
 - ▶ increased data coupling between applications => increasing maintenance burden
 - ▶ inability to access important behavior (data validation, critical business rules...)
 - ▶ need to write significant data cleansing/formatting code for poorly designed data-increased data coupling between applications

Application-interface integration

- ▶ leverage the application programming interfaces (APIs) exposed by your applications to access both the data and the functionality encapsulated by legacy systems
 - ▶ Ex: C-APIs that you can access via Java Native Interface (JNI) code (SAP, PeopleSoft)
- ▶ **But:**
 - ▶ In-house software rarely has a defined API
 - ▶ APIs may be limited in scope / may not offer the behavior that you need (or in a manner that you need it)
 - ▶ APIs are often function-oriented in nature and not object-oriented

Method-level integration

- ▶ Business logic is shared as a collection of shared methods, or operations that your software can invoke
 - ▶ Ex.: update customer data, validate a credit card transaction, deposit money into a bank account
- ▶ Provides fine-grained access to common business functions
- ▶ Easy to invoke by many programming languages
- ▶ But:
 - ▶ Fine-grained implies difficult to support transactions
 - ▶ Difficult to support common services (ex: security)

User interface-level integration

- ▶ accessing existing applications through their user interfaces
 - ▶ screen scraping: user keystrokes are simulated
- ▶ Common approach for legacy integration
- ▶ But:
 - ▶ Slow
 - ▶ Changes to the legacy user interface require changes to your code

Licenza d'uso



- ▶ Queste diapositive sono distribuite con licenza Creative Commons “Attribuzione - Non commerciale - Condividi allo stesso modo 2.5 Italia (CC BY-NC-SA 2.5)”

- ▶ Sei libero:

- ▶ di riprodurre, distribuire, comunicare al pubblico, esporre in pubblico, rappresentare, eseguire e recitare quest'opera
- ▶ di modificare quest'opera



- ▶ Alle seguenti condizioni:

- ▶ **Attribuzione** — Devi attribuire la paternità dell'opera agli autori originali e in modo tale da non suggerire che essi avallino te o il modo in cui tu usi l'opera.
- ▶ **Non commerciale** — Non puoi usare quest'opera per fini commerciali.
- ▶ **Condividi allo stesso modo** — Se alteri o trasformi quest'opera, o se la usi per crearne un'altra, puoi distribuire l'opera risultante solo con una licenza identica o equivalente a questa.



- ▶ <http://creativecommons.org/licenses/by-nc-sa/2.5/it/>