COMP3017
Service Computing

Assignment deadlines reminder

- April 2nd – hot topic study journal
- April 10th – web service programming tutorial

Scalable Service Composition and Reconfiguration in Pervasive Computing Environments

Publications

Book Chapter

- Joanna Izabela Siebert, Jiannong Cao, “Scalable Service Composition in Pervasive Computing Environments,” In Scalable Computing and Communications: Theory and Practice, S.U. Khan, L.Wang, A.Y. Zomaya (Eds.), John Wiley \& Sons, Ltd., 2013.
- Jiannong Cao, Joanna Izabela Siebert, “Service Management in Pervasive Computing Environments,“ In Pervasive Computing and Networking, M.S. Obaidat, M.Denko, I.Woungang (Eds.), John Wiley \& Sons, Ltd., 2011.

Journal Paper

- Joanna Izabela Siebert, Jiannong Cao, Steven Lai, Peng Guo, and Weiping Zhu, “LASEC: A Localized Approach to Service Composition in Pervasive Computing Environments,” submitted to IEEE Transactions on Parallel and Distributed Systems.

Conference Paper

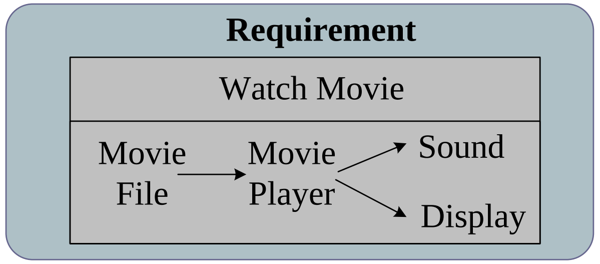
- Joanna Izabela Siebert, Jiannong Cao, Long Cheng, Edwin Wei, Canfeng Chen, and Jian Ma, “Decentralized Service Composition in Pervasive Computing Environments,“ In Proceedings of the 6th International Wireless Communications and Mobile Computing Conference (IWCMC '10), pp. 1258-1262, June 28-July 02, 2010. Caen, France.
- Joanna Izabela Siebert, Jiannong Cao, "Service Composition in Pervasive Computing Environments: a Survey,“ In Proceedings of 3rd International Interdisciplinary Technical Conference of Young Scientists, May 19-21, 2010. Poznan, Poland.
- Joanna Izabela Siebert, Jiannong Cao, Yu Zhou, Miaomiao Wang, and Vaskar Raychoudhury, “Universal Adaptor: A Novel Approach to Supporting Multi-protocol Service Discovery in Pervasive Computing,”, In Proceedings of International Conference on Embedded and Ubiquitous Computing (EUC’07), pp. 683-693, December, 2007, Taipei, Taiwan.

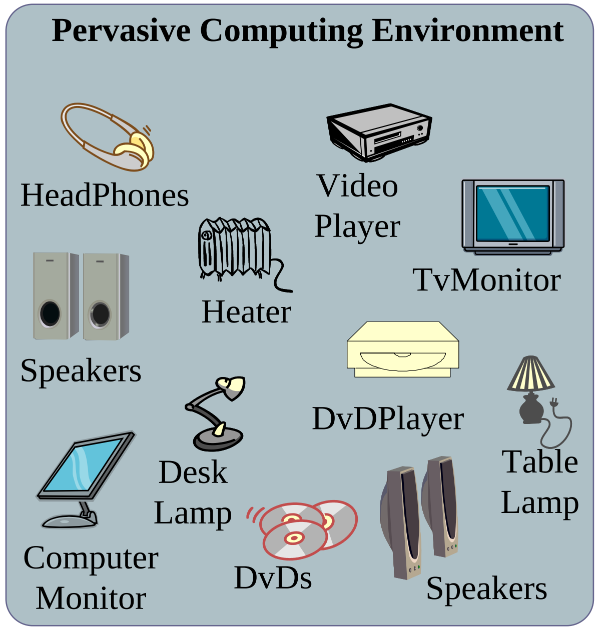
Outline

- Overview of Service Composition and Reconfiguration in PvCEs
- LASEC: Localized Approach to Service Composition
- LASER: Localized Approach to Composed Service Reconfiguration
- LASEH: Localized Approach to Service Heterogeneity
- Conclusion and Future Work

UIO-based PvCEs

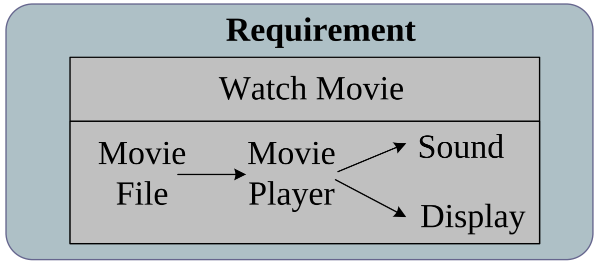
- Pervasive Computing Environment
  - A medium that provides user with all the functionality he needs to satisfy his requirements. It is built on physical world with embedded computing devices.
- UIOs
  - Smart objects augmented with various processing capabilities
  - Mirrors can help with shopping
  - Objects at home can serve as interfaces for controlling robots
  - Drinking cup can monitor and remind about healthy drinking habits

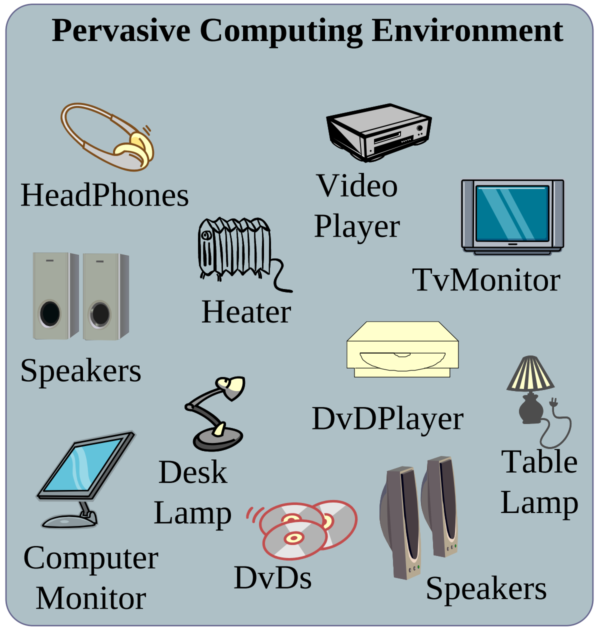




Service Composition in PvCEs (1)

- Service
  - A functionality of a computational entity whose execution satisfies the requestor’s requirement.
- Service Composition
  - A process of identifying and combining component functionalities to compose a higher level functionality and provide means to perform the requested functionality.





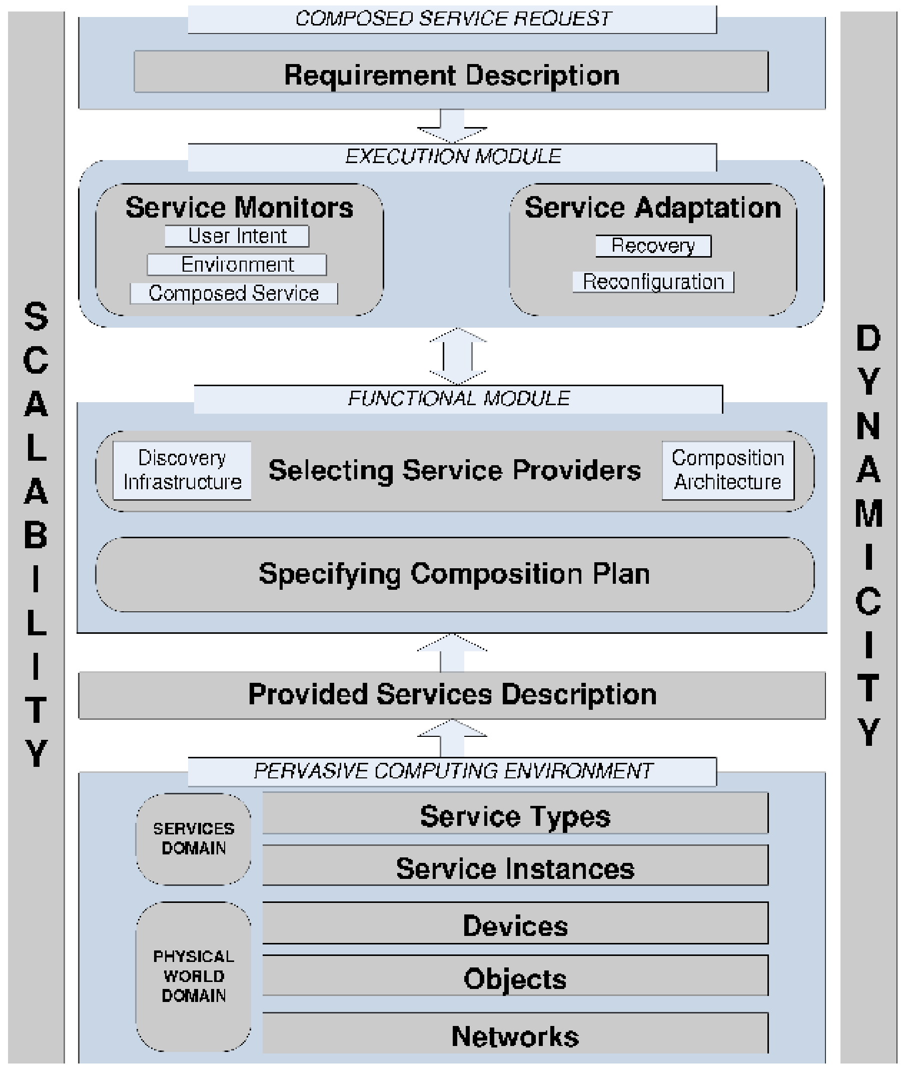
Service Composition in PvCEs (2)

- Characteristics of the UIO-based PvC environment
    - Service requestors require services instantaneously to accomplish their goals
    - Service requestors have no prior knowledge about the available services
    - Service providers join and leave environments at run-time
    - The demand of service requestors changes together with changes in context

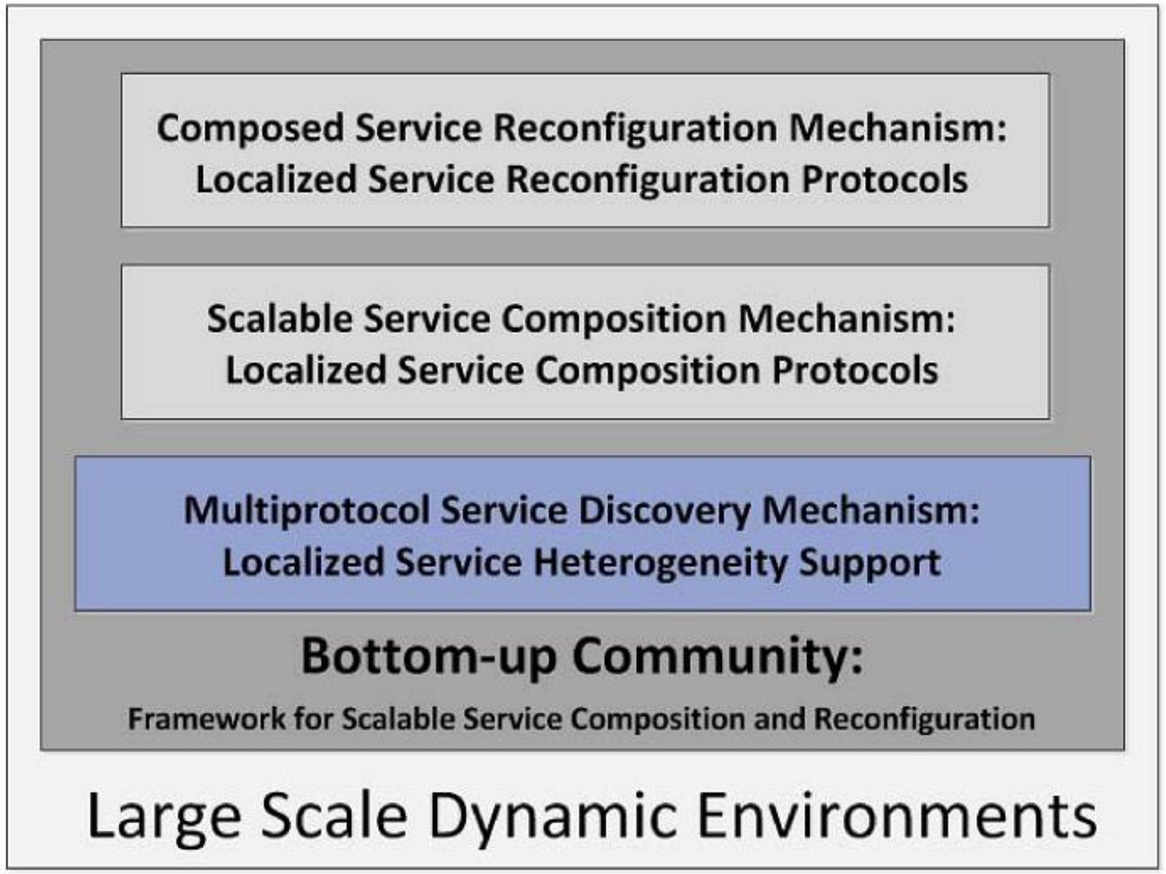
Key Issues for Service Composition and in PvCEs

- Bottom-up UIO collaboration support
  - In UIO based PvCE, there is very large number of resource constrained UIOs and no guarantee of having a central entity capable of managing all other devices.
  - To achieve desirable system behaviors, autonomous interaction of UIOs is required - enable to coordinate the UIOs and make them serve people in a less obtrusive manner.
- Handling changes in environments
  - Assist people with computation support everywhere and all the time.
  - PvCE is extremely dynamic in nature and UIOs frequently join and leave the environment.
- Heterogeneity support
  - Heterogeneity in computing systems will not disappear in the future, but instead will increase as the range of computing devices widens.
  - In UIO based PvCE heterogeneity is particularly high.

Service Composition Framework



Framework for Scalable Service Composition and Reconfiguration


Composed Service Reconfiguration Mechanism:
Localized Service Reconfiguration Protocols
Scalable Service Composition Mechanism:
Localized Service Composition Protocols
Multiprotocol Service Discovery Mechanism:
Localized Service Heterogeneity Support
Bottom-up Community:
Framework for Scalable Service Composition and Reconfiguration
Large Scale Dynamic Environments

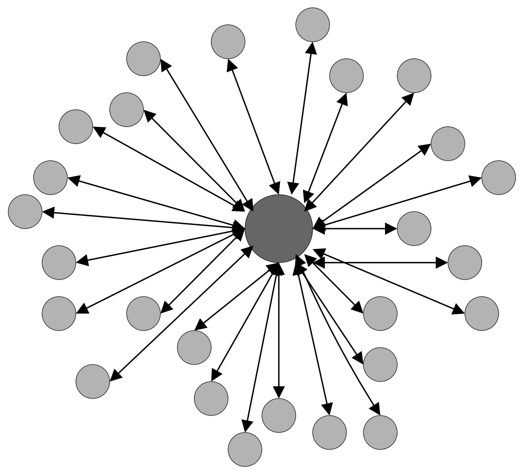
LASEC: Localized Approach to Service Composition

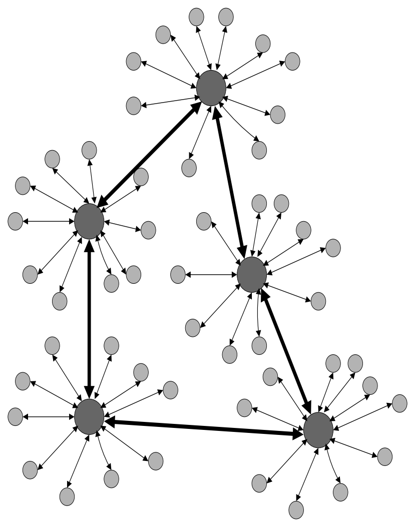
Your task

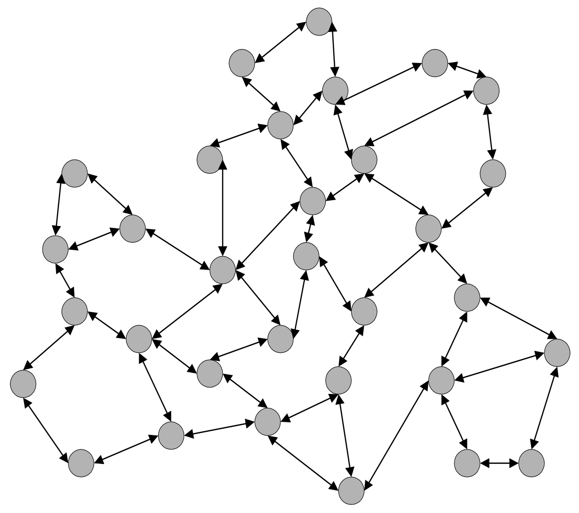
- When listening to the sharing about this research, please keep in mind these questions:
  - What are the difference between this approach to discover the relevant service providers and the approach used in Web Services?
  - In what scenarios UDDI can be better and when LASEC will have more advantages?

Background

- Existing approaches
  - Centralized - as the service composition environment may be dynamic and large scale, centralized service composition algorithm is usually inefficient due to message cost.
  - Decentralized with coordinators - a decentralized approach, which employs pre-determined coordinators to search and compose service, may have high cost as well, if the coordinators are far from those devices that should be composed.


Centralized


Hybrid

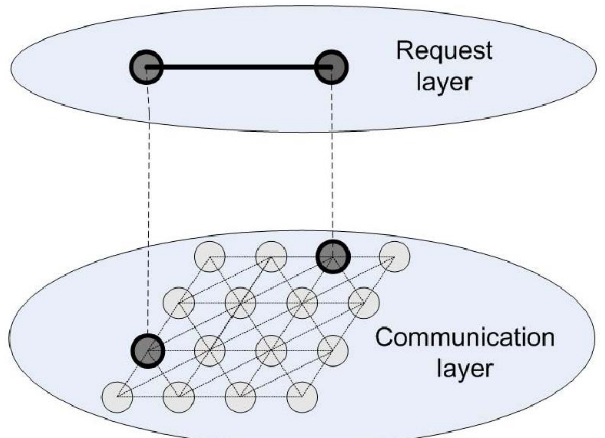

Decentralized

Motivation

- Common feature - globalized nature of algorithms
  - At least one node needs to maintain global information about the current task and about network status.
  - Unsuitable for highly dynamic and large scale environments - huge communication overhead, especially with frequent changes of topology.
  - Global knowledge is not always necessary, since in service composition for PvCE, it is beneficial when the distance between the services used in composition is as small as possible.
- To remedy the problem of maintaining global information, we have opted to adopt a localized approach, which naturally tends to select compositions with high degree of composition locality.

Problem Statement

- Given
  - A set of nodes each providing certain services, the nodes are interconnected and can communicate with each other via wireless network,
- Assume
  - A user specified requested composite service with specified types of requested services and relationships between them.
- Objective
  - Design a localized protocol for the nodes to collaborate in the composition process
- Such that
  - The communication cost is minimized.

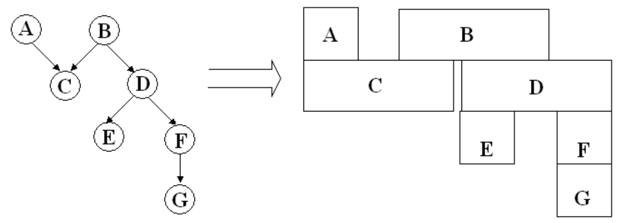


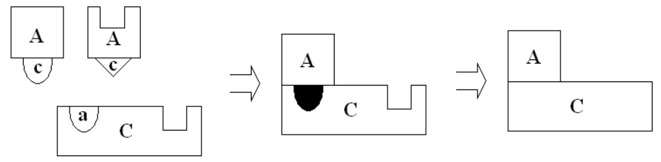
Overview of the Solution (1)

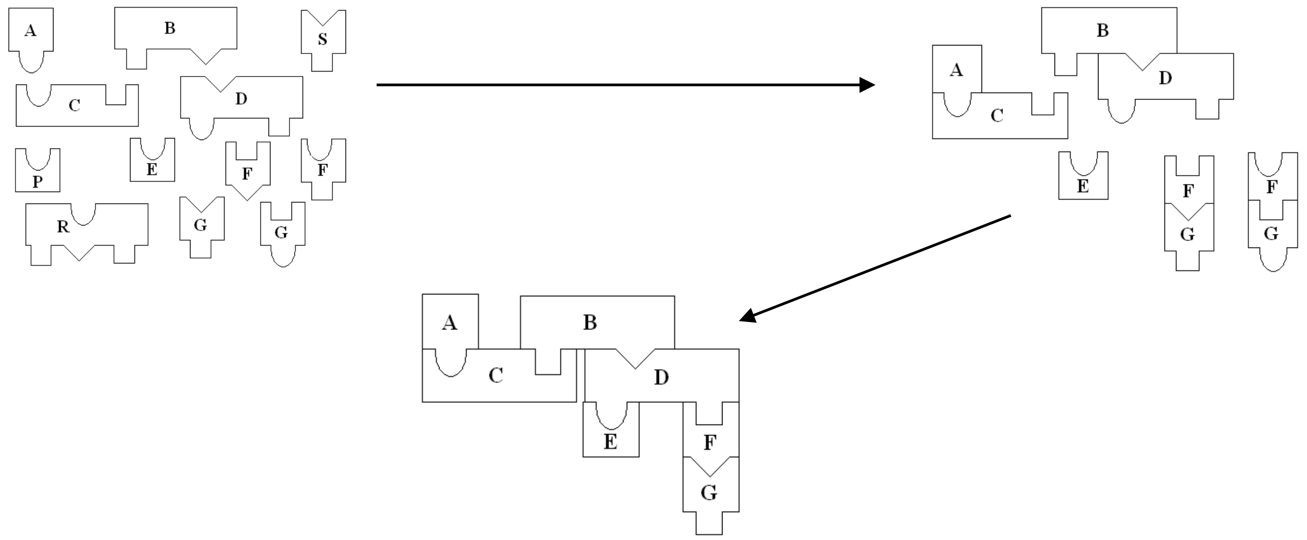
- Requirements for new solution
  - There should be no special entity to manage service composition process
  - Service providers can communicate only with their local neighbours, not all other service providers
  - No service provider knows the full global information or gathers it
  - Initially, service providers are atomic, in sense that they provide some functionality and are not coupled with any other service provider to provide a composed service

Overview of the Solution (2)

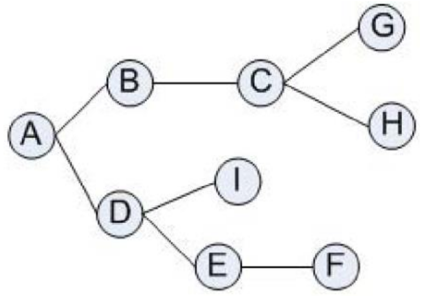
- Inspired by the strategies adopted in solving jigsaw puzzles.
- Forming service sections: Service provider identifies what service type it needs to interlock with through his output and then searches for appropriate service provider with matching input type.
- Growing service: At later stages these sections are joined together with other sections to gradually construct the completed solution.

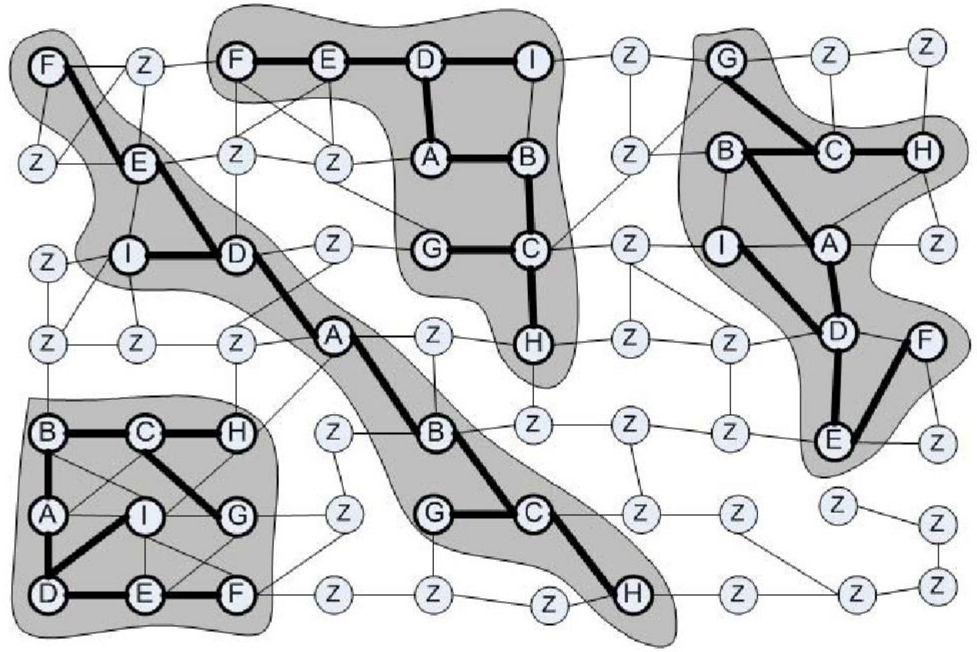


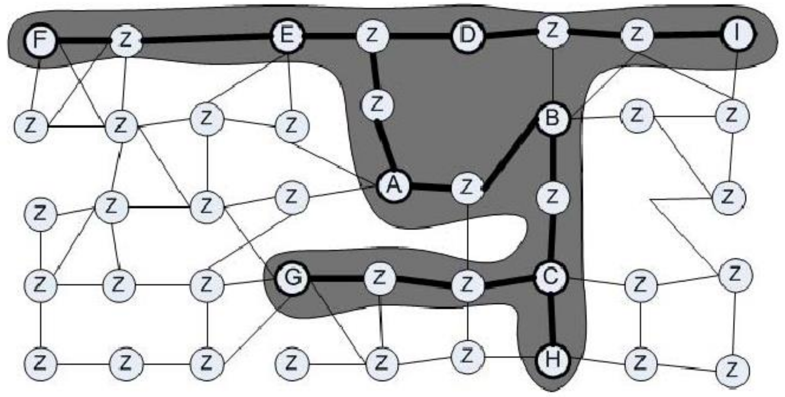




Overview of the Solution (3)

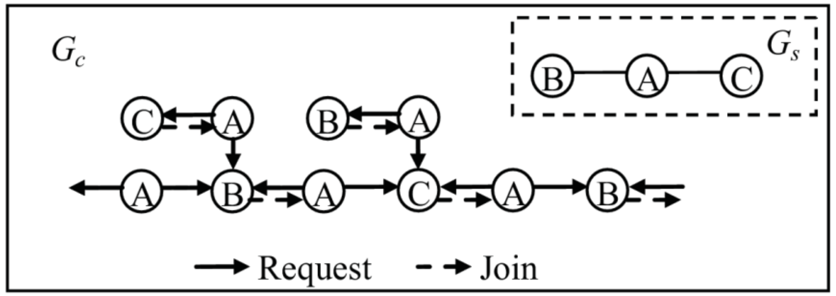






Challenging Issues

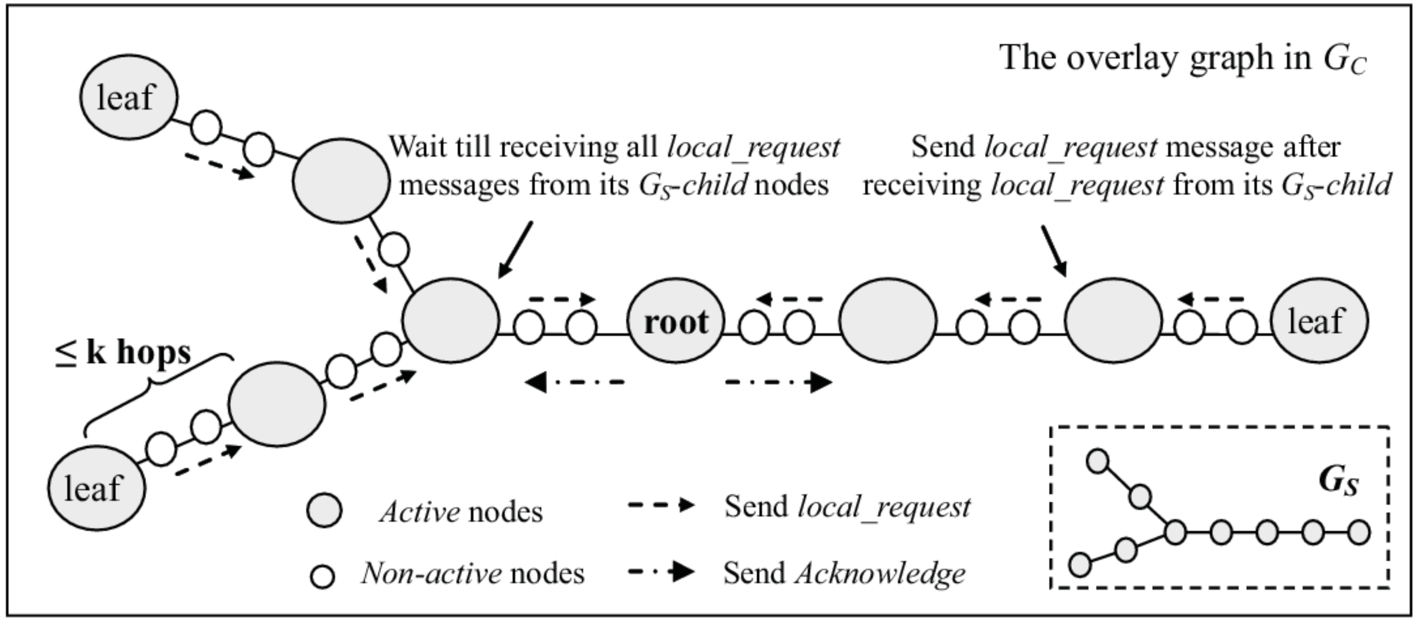
- Awareness
  - How does a node know the current members of the section that it has participated in? How does a node know the section that it has participated in has already failed?
- Judgment
  - How does a node know which section is more possible to succeed in the composition when the node is required to join? How to reduce or stop the useless composition which cannot be completed?
- Assembly
  - Since there is no coordinator in the network, how to guarantee the composition to converge into one integrated overlay graph.



How to constraint the blind and random composition among the UIOs that could not form the requested service so as to guarantee all the services specified in the request will be provided

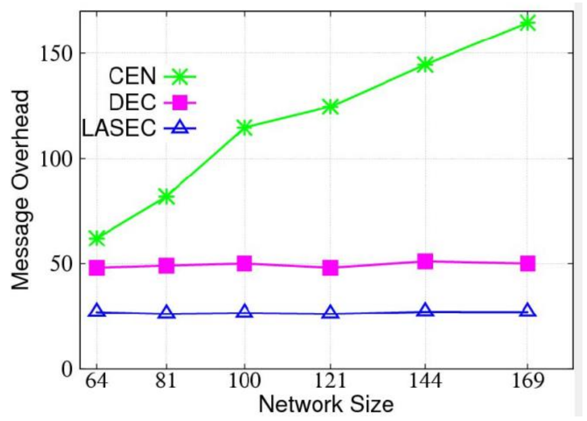
Basic Idea of LASEC

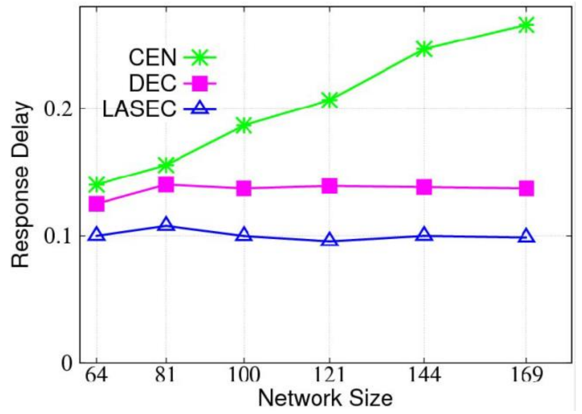
- A-Ack mechanism
  - UIO decide on collaborating with another UIO only after obtaining some additional information from the collaboration candidate.
  - Specifically this information refers to ability of given UIO to compose another part of the service.


The overlay graph in G_C
leaf
Wait till receiving all *local_request* messages from its G_S-*child* nodes
Send *local_request* message after receiving *local_request* from its G_S-*child*
root
leaf
≤ k hops
leaf
Active nodes
Non-active nodes
Send *local_request*
Send *Acknowledge*
G_S

Simulation Results (1)

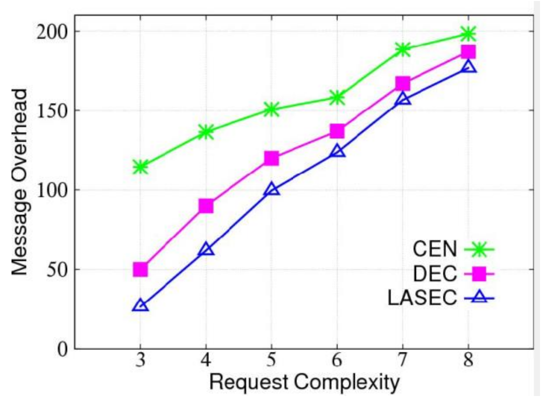
- **Scalability with regards to network size**
- **Message overhead**
  - Our LASEC service composition mechanisms require less messages than CEN for finding first result.
  - LASEC scales very well with the growing size of the network while CEN imposes larger cost when the network grows.
- **Response delay**
  - LASEC algorithm performs better than DEC and CEN.
  - Similar as with message cost, LASEC and DEC scale better than CEN.

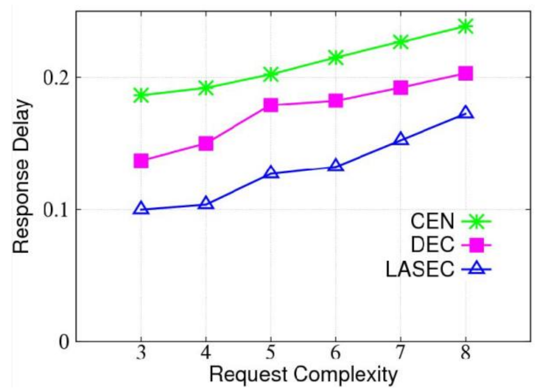




Simulation Results (2)

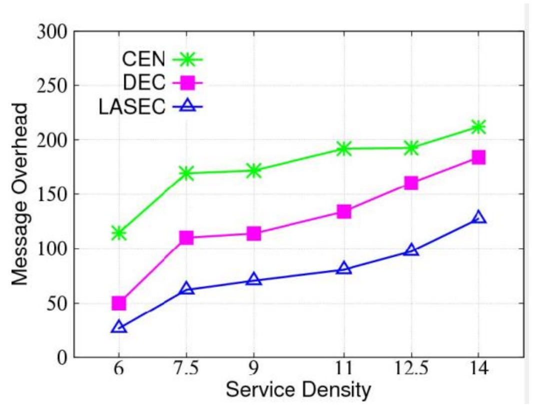
- **Scalability with regards to request complexity**
- **Message overhead**
  - Number of messages increases with the complexity of the request.
  - Along with the growth of request complexity, LASEC and DEC generate more messages, while CEN is more scalable.
- **Response delay**
  - Our LASEC service composition mechanism performs better than DEC and CEN for finding first result.
  - Moreover, DEC, CEN and LASEC perform similar in terms of scalability.

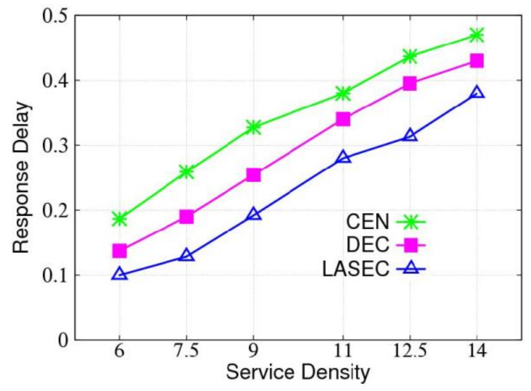




Simulation Results (3)

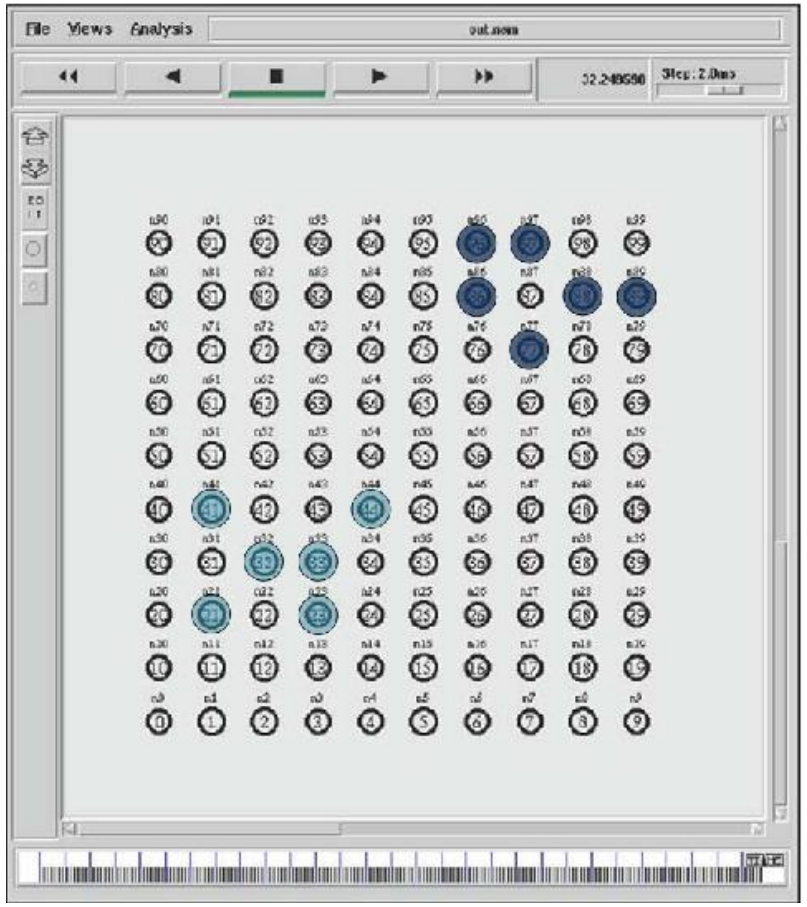
- **In these simulations we have explored localized characteristics of our approach and allowed nodes to forward message to 2 hop neighbours only.**
- **We observe that response delay as well as number of messages increase with service density.**
- **Our LASEC service composition mechanism has smaller overhead than DEC and CEN for finding the result**





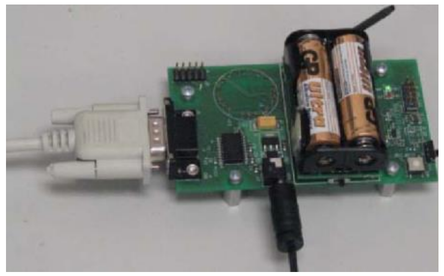
Simulation Results (4)

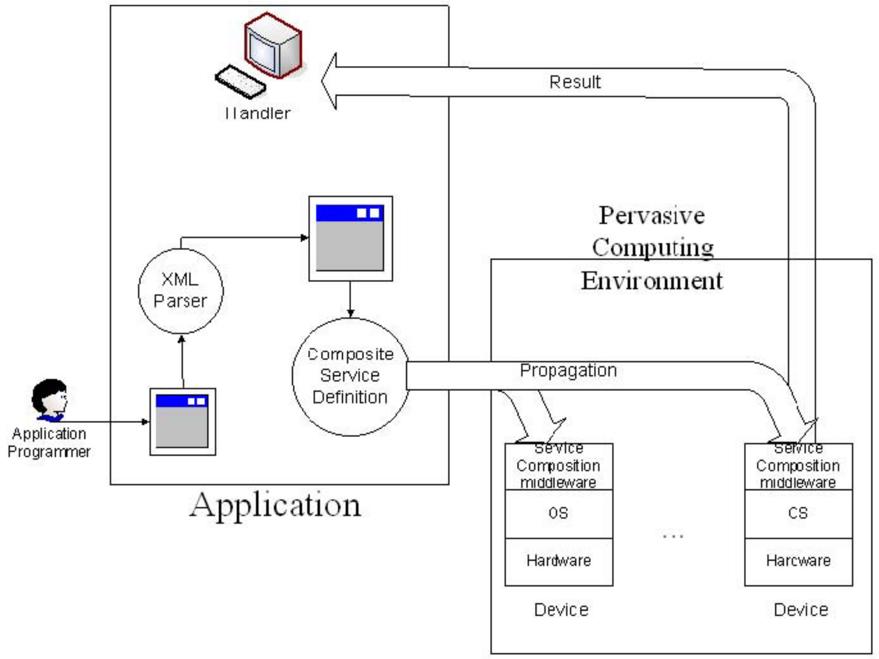
- **Composition locality - the distance between the services used in composition**
- **LASEC tends to select compositions with high degree of composition locality**



Experiment Results (1)

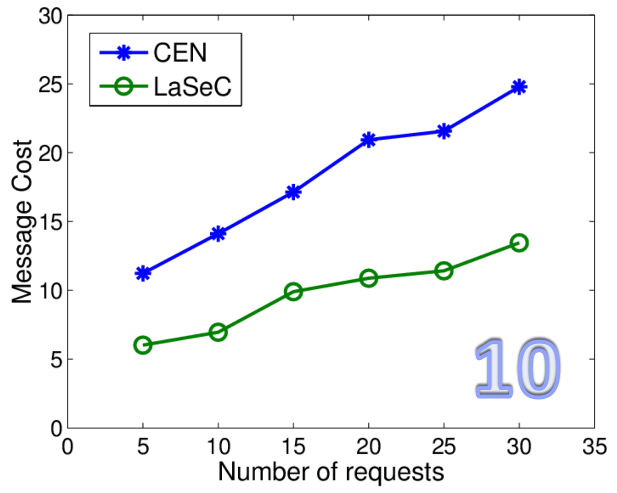
40 MicaZ nodes with a MIB600 gateway deployed in an indoor environment.

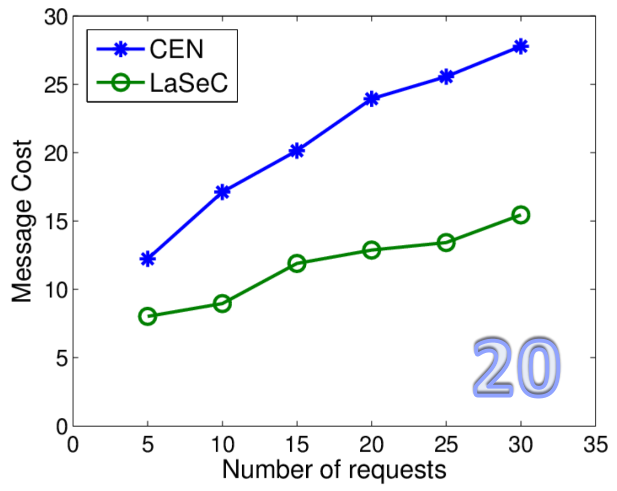



Handler
Result
Pervasive Computing Environment
XML Parser
Composite Service Definition
Propagation
Application Programmer
Application
Sevice Composition middleware
OS
Hardware
Device
...
Service Composition middleware
CS
Harcware
Device

Experiment Results (2)

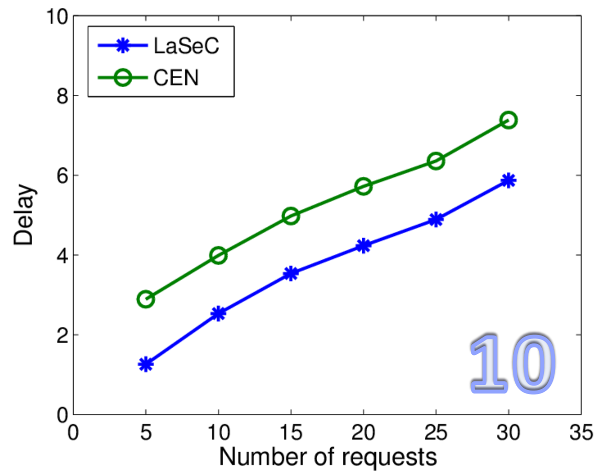
- **Message cost**
  - LASEC can reduce the message cost by over 50 percent
  - LASEC scales better when the number of services increases

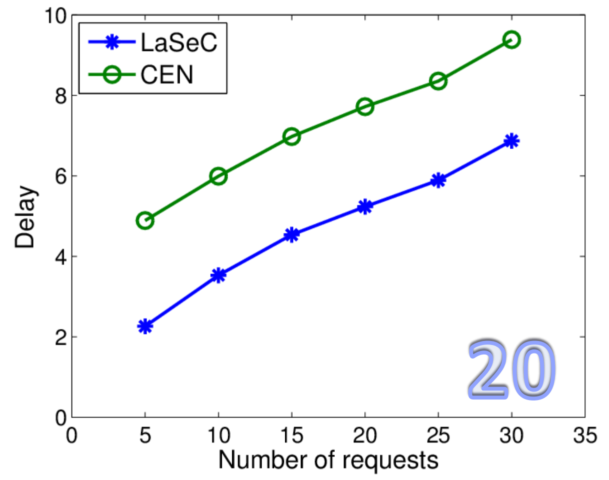




Experiment Results (3)

- **Message delay**
  - LASEC achieves smaller delay, especially when there are more services in the environment.





Open Question Points: 10

Setting

In the research just introduced to you, a novel approach was proposed, in which service providers collaborate in service composition process with each other. When the request for the composed service comes in to the system, service providers start to exchange messages following the proposed LASEC algorithm and gradually find the partners with which they can provide the composed service to the user.

What are the difference between this approach to discover the relevant service providers and the approach used in Web Services?

Answer

LASEC vs UDDI

- UDDI is a globalized algorithm
  - UDDI nodes need to maintain global information about the service providers.
- LASEC is a localized algorithm
  - Service providers can communicate with their local neighbours when the request comes in to the system, there is no special registry that would maintain the information about service providers

Open Question Points: 10

Setting

In the research just introduced to you, a novel approach was proposed, in which service providers collaborate in service composition process with each other. When the request for the composed service comes in to the system, service providers start to exchange messages following the proposed LASEC algorithm and gradually find the partners with which they can provide the composed service to the user.

In what scenarios UDDI can be better and when LASEC will have more advantages?

Open Question is only supported on Version 2.0 or newer.

Answer

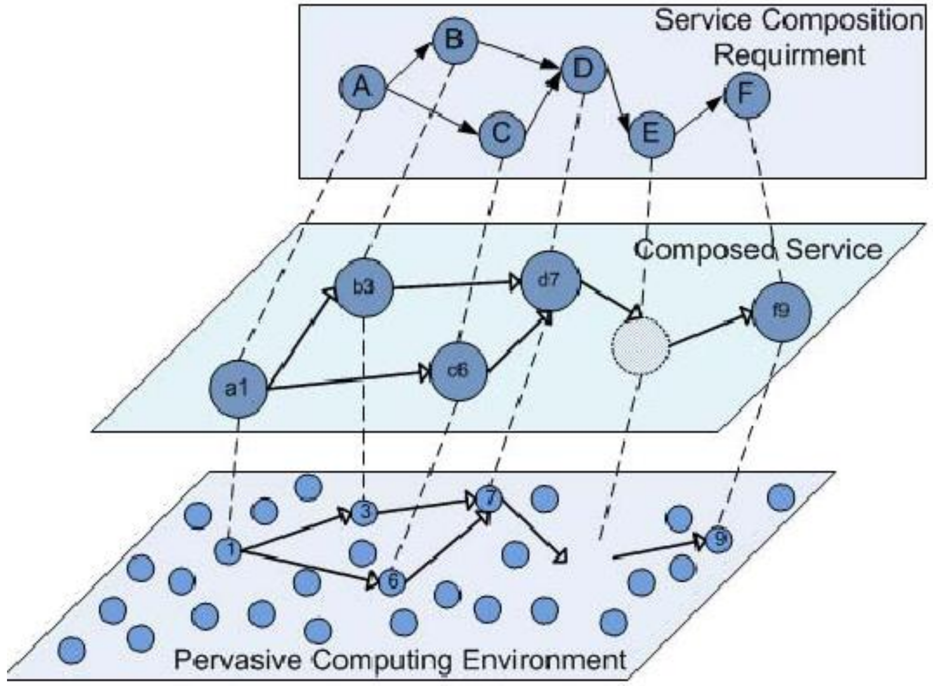
LASEC vs UDDI

- Consider the dynamicity of the environment
    - UDDI is a good option when the environment is very stable, there are not many changes
    - LASEC offers more advantages in dynamic environments:
        - Service providers join and leave environments at run-time
        - The demand of service requestors changes together with changes in context

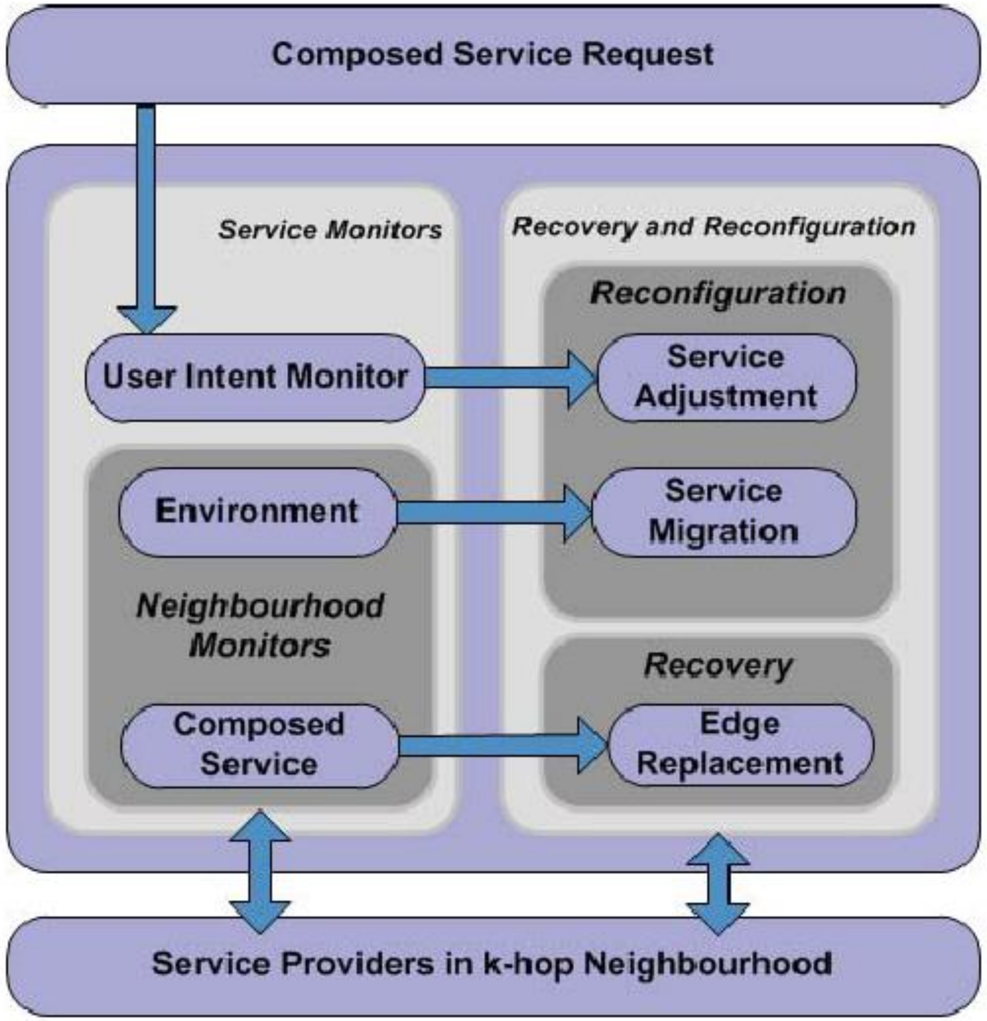
LASER: Localized Approach to Composed Service Reconfiguration

Background

- Even when once the initial service composition is established, the change in the composed solution may be required
  - Mobility/failure of service providers in the original service composition
  - Better services available
  - Change in the requirement


Service Composition Requirment
A
B
C
D
E
F
Composed Service
a1
b3
c6
d7
f9
Pervasive Computing Environment
1
3
6
7
9

System Model

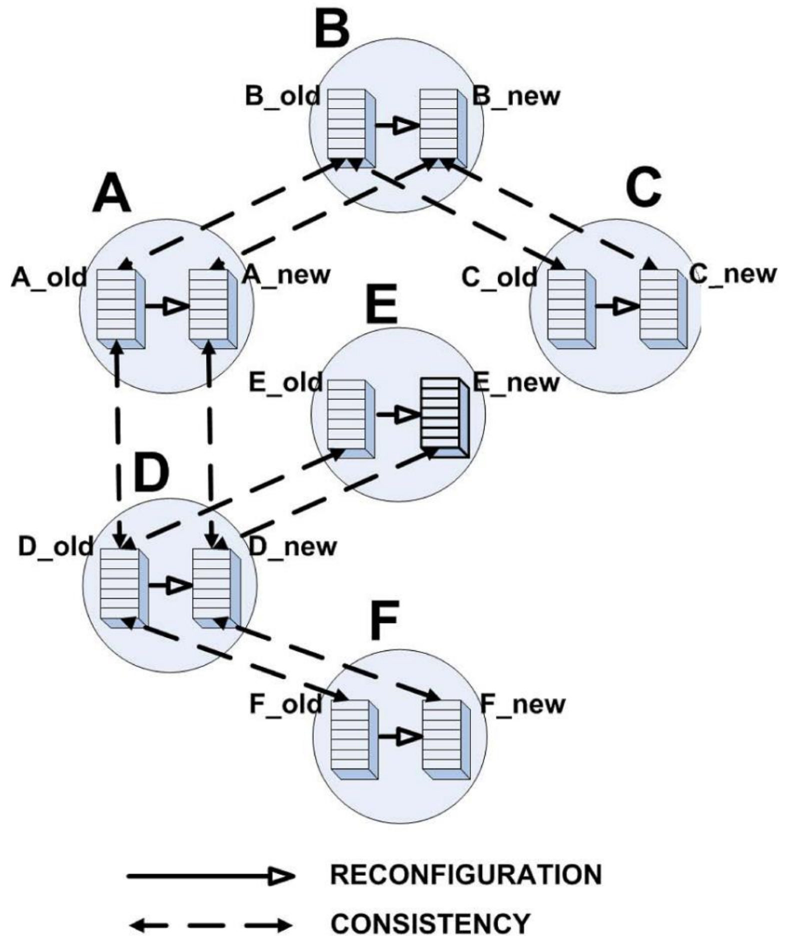


Consistency Requirements

- Major reconfiguration requirement is preservation of composed service consistency
  - The service providers in the composed service are in mutually consistent states
  - The composed service satisfies its composition reliability requirements

Mutually Consistent States

- Transform composed service with service providers in mutually consistent states into new composed service that maintains this mutual consistency.



Composition Reliability

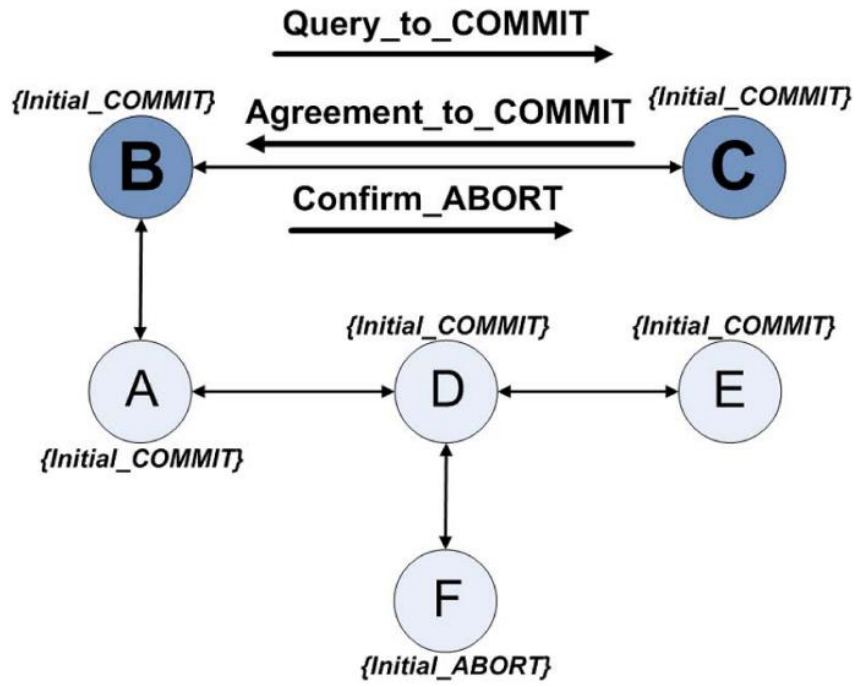
- Configure composed service from one consistent state to another providing following properties:
  - Atomicity - the changes during service recovery are atomic: either all operations that are part of the recovery occur or none occurs.
  - Consistency - service recovery reconfigures composed service between consistent states.
  - Isolation - even though service recoveries can be executed concurrently, no service recovery sees another service recovery work in progress. The service recoveries seem to run serially.
  - Durability - after a service recovery completes successfully, its changes survive subsequent failures.

Maintaining ACID Properties

- Recoverable services
  - Recoverable services log their actions and therefore can restore earlier states if a failure occurs.
- A commit protocol
  - A commit protocol allows multiple service providers to coordinate the committing or aborting of partial executions of the composed service.

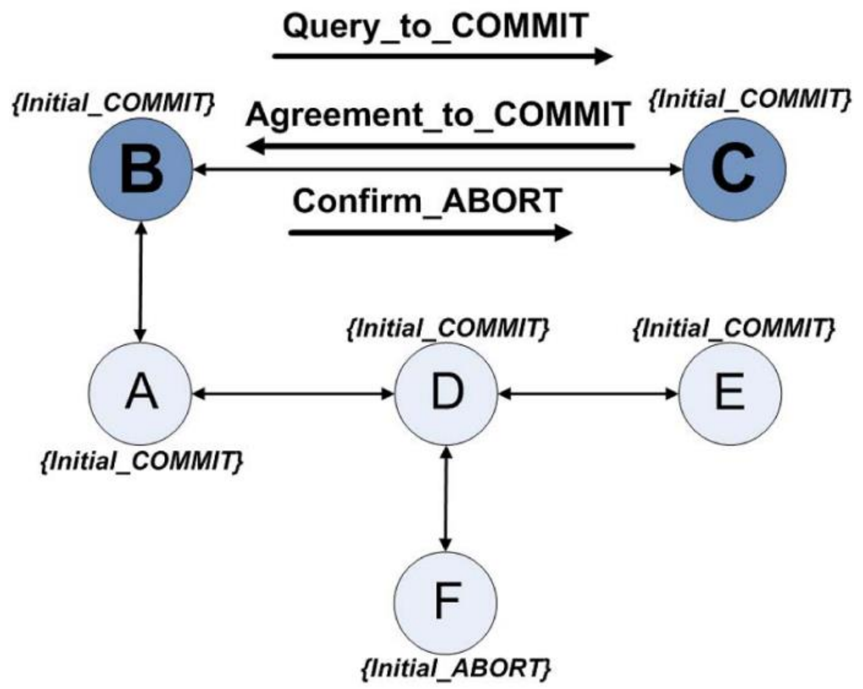
Problem of Hidden Abort

- Existing commit protocols for distributed transactions require coordinator to collect global information and make abort/commit decision
- In localized system without coordinator commit protocol may result in hidden abort



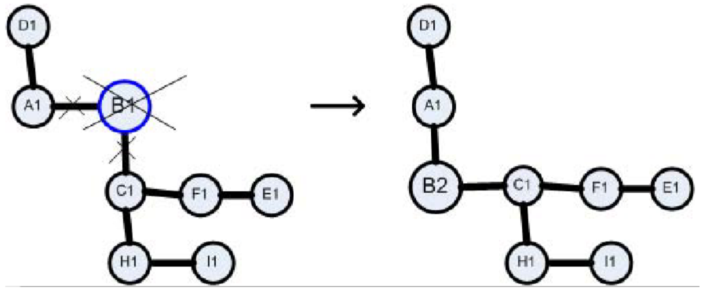
Gradual Reveal

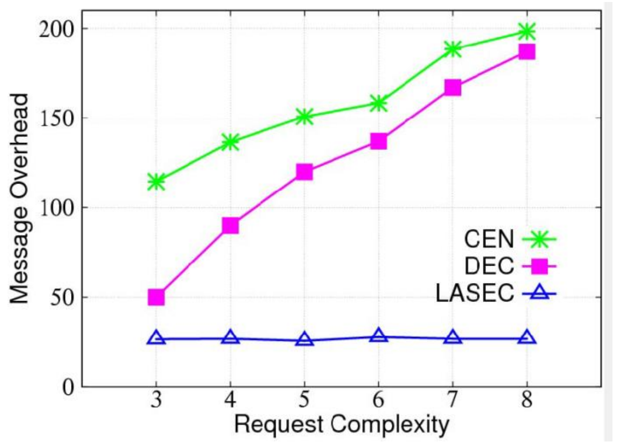
- **Helps nodes to avoid making premature decisions on commit**
- **Couples vote and completion phases of commit protocol**
- **Abort in distant node gradually reveals itself to the interested nodes**

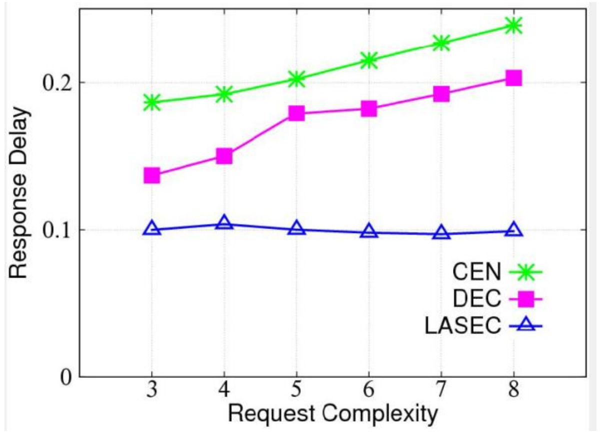


Simulation Results (1)

- **Coping with request dynamicity**
  - Recomposition cost is stable and similar to that of composing simple requests.

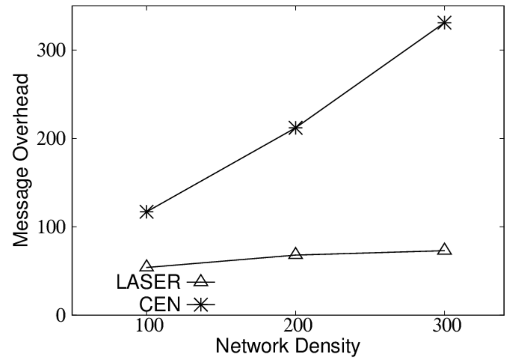


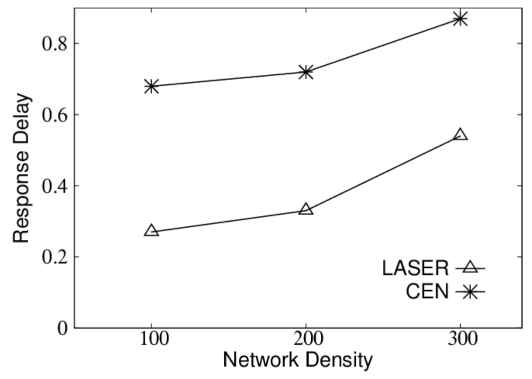


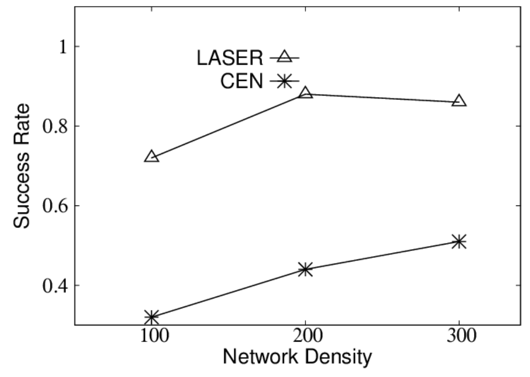


Simulation Results (2)

- **Under mobility conditions**
- **Message overhead**
  - The message overhead for LASEC is not very high. It also increases with the increase in network density, but not as much as CEN.
- **Response delay**
  - Increases with the increase in network density.
- **Success rate**
  - In LASER, success rate is high and initially increases with network density and then decreases with further increase in network density (high recovery delay may render the recovery unsuccessful)







LASEH: Localized Approach to Service Heterogeneity

Background

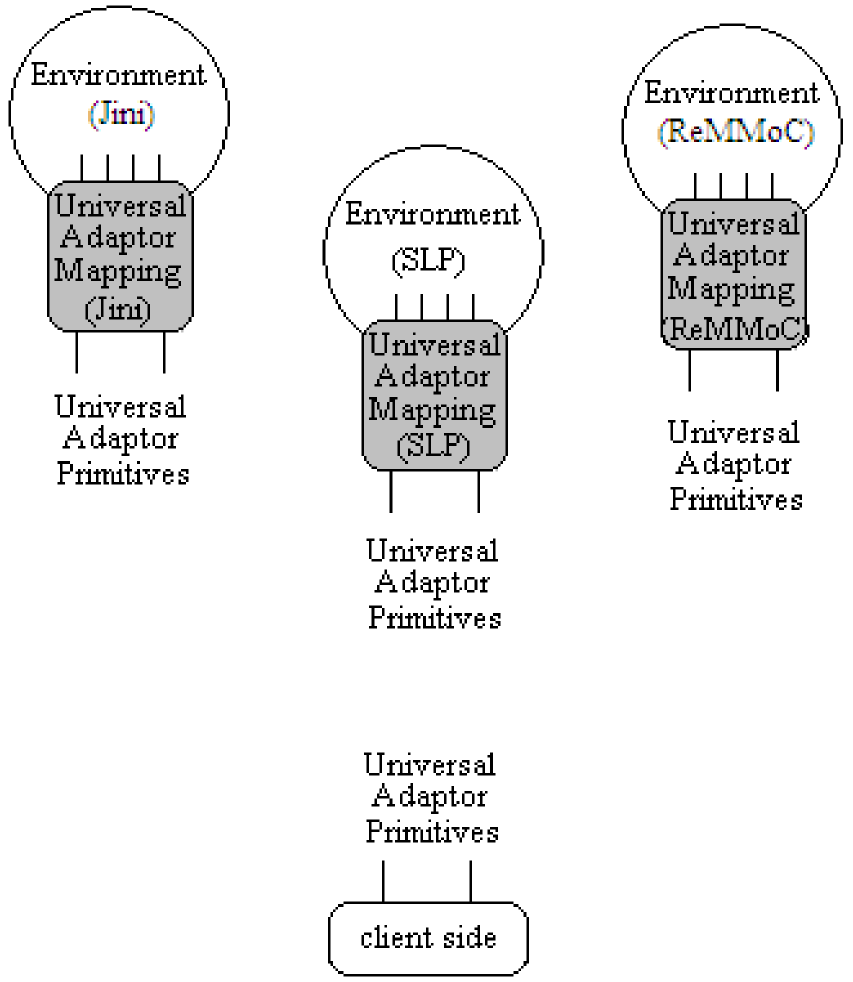
- Heterogeneity - different environments supported by different service management systems, which may use standard protocols as well as tailored made mechanisms
  - software,
  - hardware devices,
  - network infrastructures
- Diverse service discovery mechanisms

How to integrate or bridge these service discovery schemes ?

Interoperability Requirements

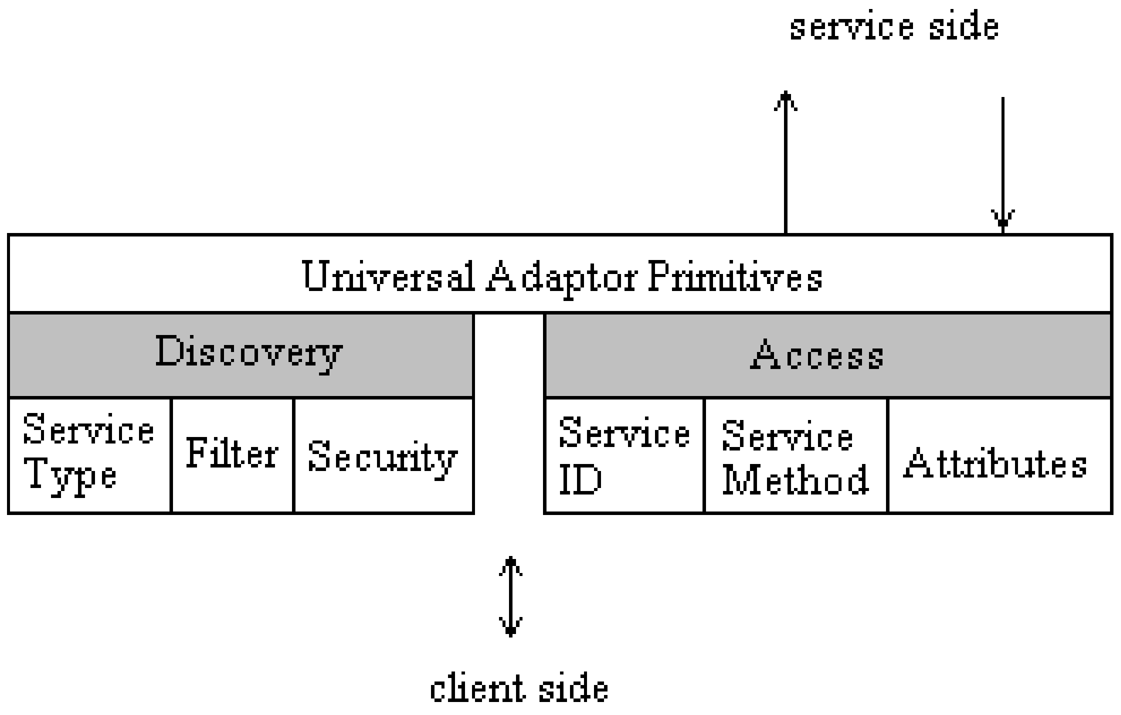
- No change should be imposed on the existing service discovery mechanisms
- No change should be imposed on the services registered in domains
- No functionality of the environment should be compromised
- The system should be lightweight, scalable, and extendable
- Support both standard and tailor-made service discovery mechanisms

System Model and Architecture



- **Environment**
  - Service Discovery Domain, where a native service discovery system is able to find a specified resources if they are available in the domain.
- **Services**
  - provided by hardware devices, software, and other entities
  - Examples: weather forecast, stock quotes, and language translation. - provided directly or via another device

Universal Adaptor Primitives (UAP)



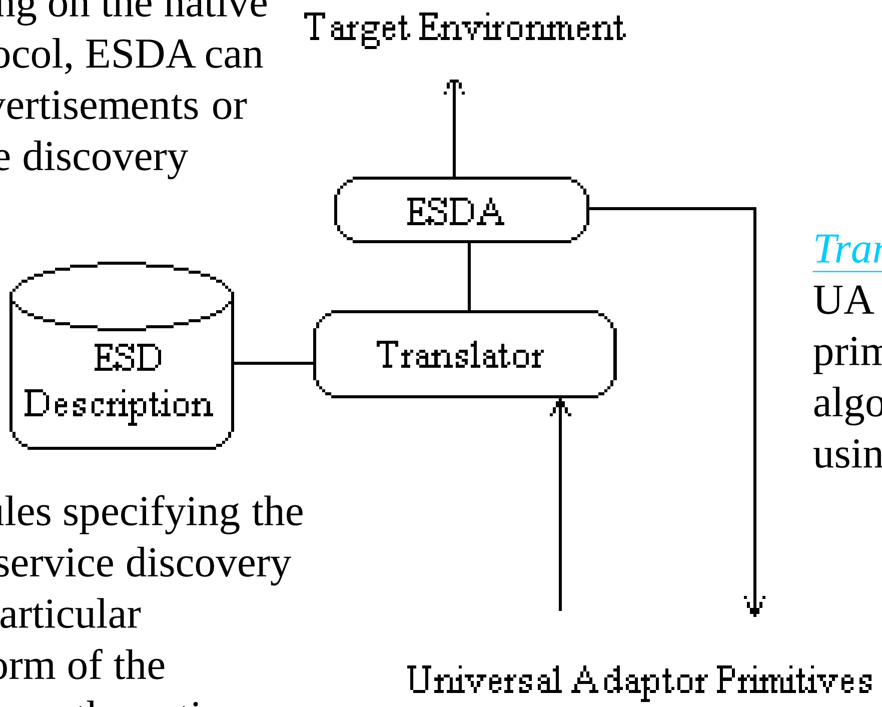
[Return_Values] Primitive_Name [Parameters]

[RV_ST, RV_F, RV_S] Discovery (Service Type, Filter, Security)

[RV_status] Access (ServiceID, Service Method, Attributes)

Universal Adaptor Mapping (UAM)

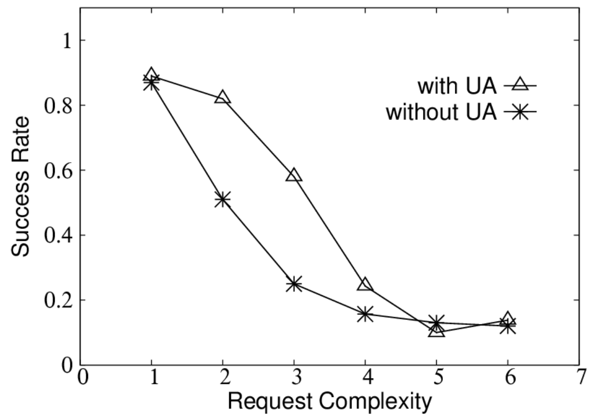
ESDA - Agent that uses the algorithm generated by Translator to perform service discovery in the target environment. Depending on the native service discovery protocol, ESDA can register for service advertisements or performs active service discovery

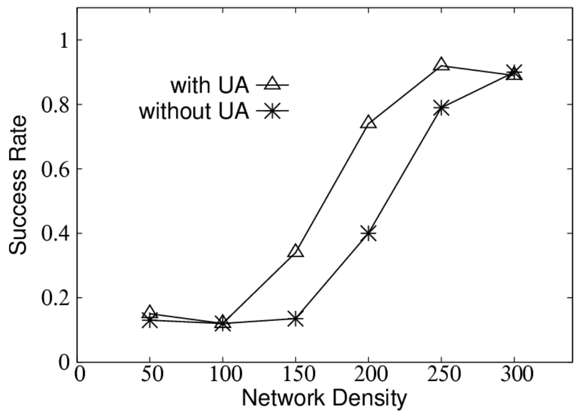


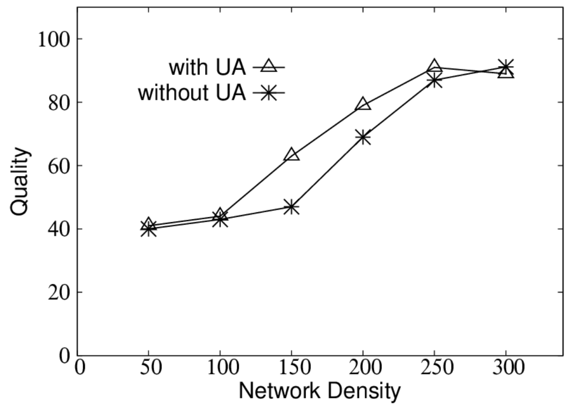
Translator - Module that translates UA primitives into native primitives, and outputs an algorithm for service discovery using the native primitives

ESD Description - Rules specifying the characteristics of the service discovery protocol used in the particular environment, in the form of the primitives extracted from the native protocol

Simulation Results



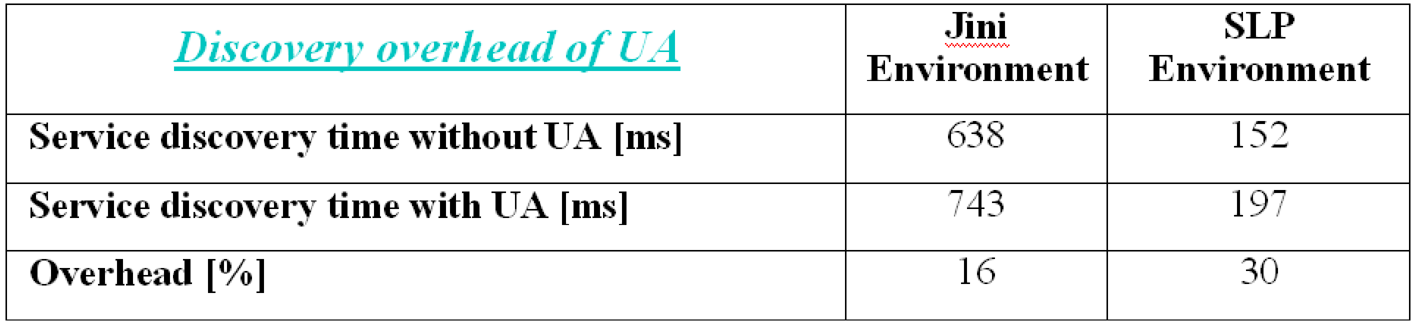




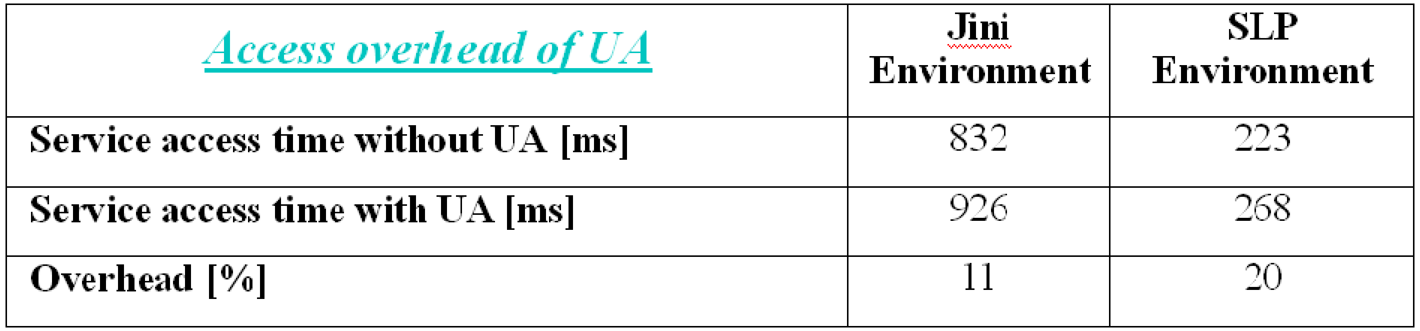
- For the too sparse network, neither approach with UA not without UA can achieve good results.
- The more neighbors the node has the bigger success rate of finding the preferred collaboration partner, quality improves as well.
- When the network is very dense, the improvement is not visible any more - additional neighbors available for the node to choose from are treated as duplication.
- Approach with UA always results with providing bigger choice to the node, which in turn gives better success rate and higher quality.
- With UA and without UA results are not very symmetric. The network for the approach with UA becomes “too dense” earlier than for the approach without UA.

Experiment Results

- **Two service discovery environments:**
  - Jini - Jini2_1
  - SLP - jSLP
- **Example: a weather information service**



Discovery overhead of UA	Jini Environment	SLP Environment
Service discovery time without UA [ms]	638	152
Service discovery time with UA [ms]	743	197
Overhead [%]	16	30



Access overhead of UA	Jini Environment	SLP Environment
Service access time without UA [ms]	832	223
Service access time with UA [ms]	926	268
Overhead [%]	11	20

Conclusion

- **We investigated the characteristics of UIO based large scale and dynamic pervasive computing environments, and identified the new challenges caused by these characteristics in the design of service composition support.**
- **We then study how to design localized service composition protocols in large scale dynamic environments.**
  - Scalable service composition
  - Composed service reconfiguration
  - Heterogeneity support
- **The bottom-up community is scalable enough to carry on with service composition operations not depending on the scale of the environments.**

Future Work

- **Evaluation**
  - Test the proposed solution with larger networks
  - Use real devices to compare results with the simulation
  - Carry out further experimentation using multitude of UIOs
- **QoS support**
  - Different approaches to increasing QoS utilizing localized interactions
  - Consider diverse parameters of quality and incorporate them into the service composition mechanism

- The content of the research presented today is not included in the final examination

Module Five: Service Oriented Architecture: architectural patterns and modelling methods of SOA, designing a distributed service system with SOA

Our materials for this module:

- **Thomas Erl, Service-Oriented Architecture, Concepts, Technology, and Design, ISBN：9787030336422，**
  - Chapter 3
  - Chapter 8
- **Online resources:**
  https://www.guru99.com/soa-principles.html

Module 5 Learning Outcomes

- Analyze common characteristics of SOA
- List common benefits of using SOA
- Assess common pitfalls of adopting SOA
- Understand the service orientation in the enterprise
- Be familiar with anatomy of a service oriented architecture
- Be able to explain the common principles of service orientation
- Understand how service orientation principles inter-relate
- Be able to evaluate Web service support for service-orientation principles

Guided questions for Module 5

- What does the “service oriented” concept mean?
- What is a service oriented architecture?
- What are services within SOA?
- What is a business process?
- How is a service related to a business process?
- How services relate to each other within SOA?
- How services communicate?
- How services are designed?

Guided questions for Module 5

- What are the principles of service-orientation?
- What are characteristics of contemporary SOA?
- What are the benefits of SOA?
- What are the pitfalls of adopting SOA?
- How service orientation applies to the enterprise
- How service orientation principles inter-relate?
- How Web services support service orientation principles?

Service Oriented Architecture

Architecture

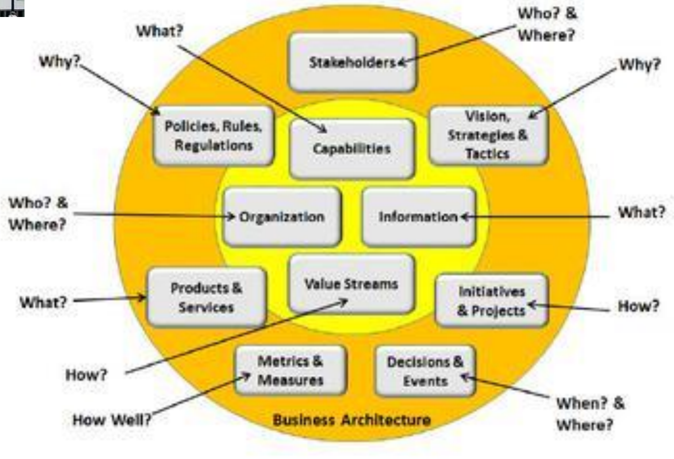
- The process of planning, designing and constructing structures

Architecture

- The process of planning, designing and constructing structures
  - Building

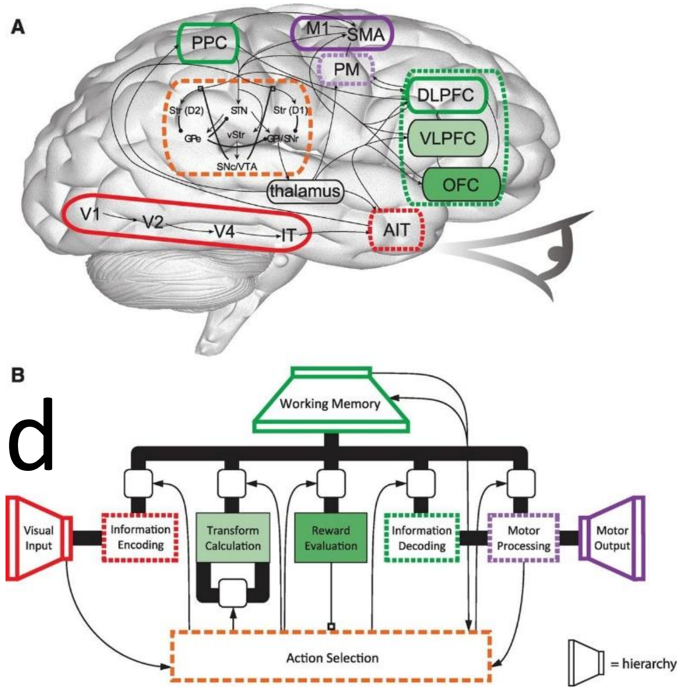
Architecture

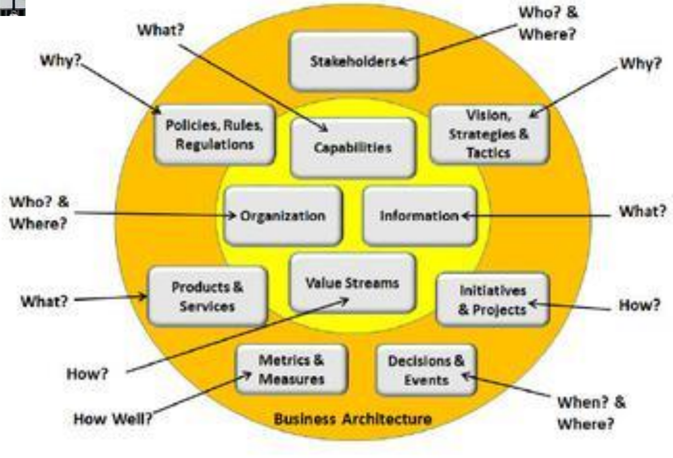
- The process of planning, designing and constructing structures
  - Building
  - Business



Architecture

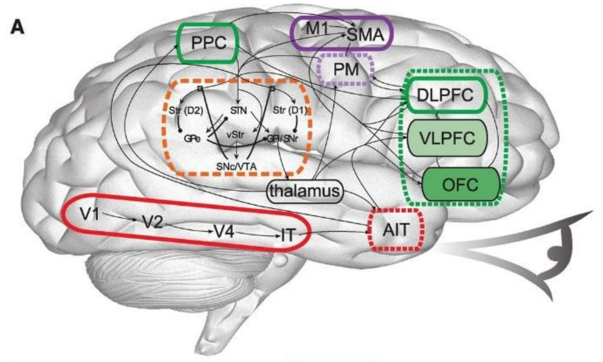
- The process of planning, designing and constructing structures
  - Building
  - Business
  - Cognitive

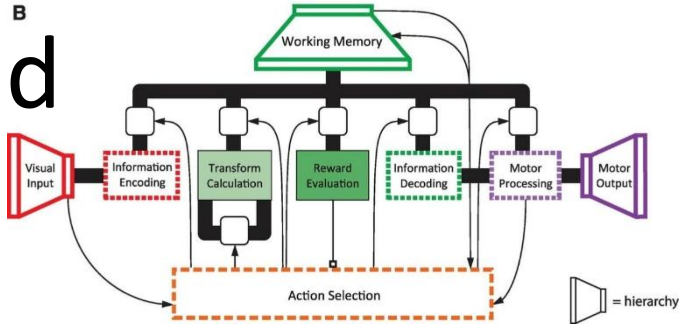


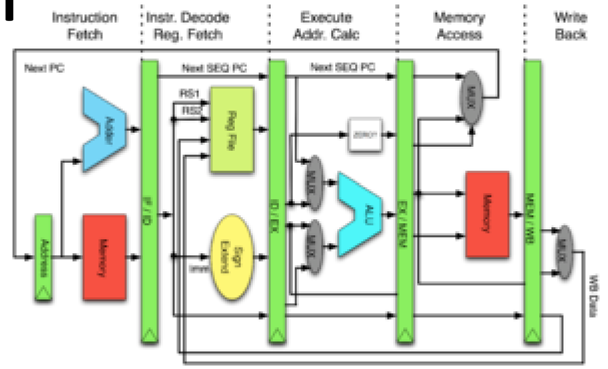


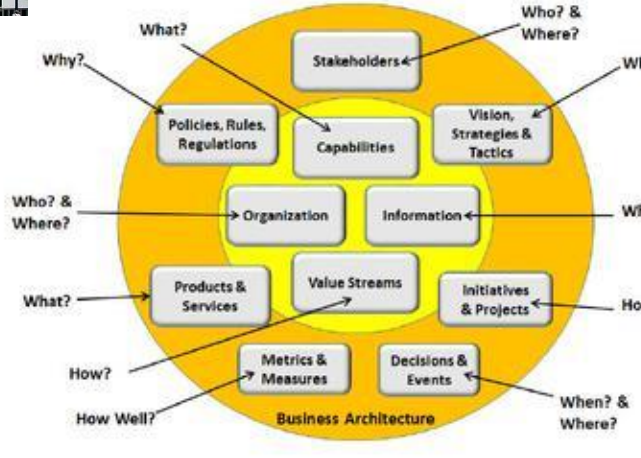
Architecture

- The process of planning, designing and constructing structures
  - Building
  - Business
  - Cognitive
  - Computer

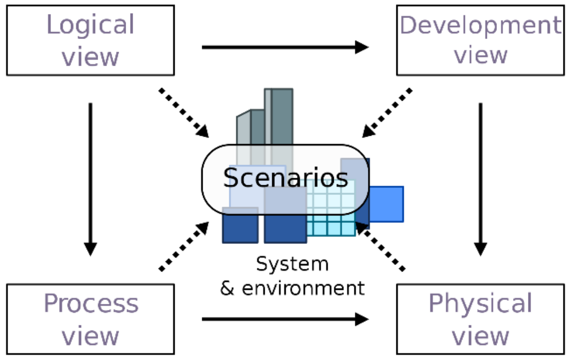


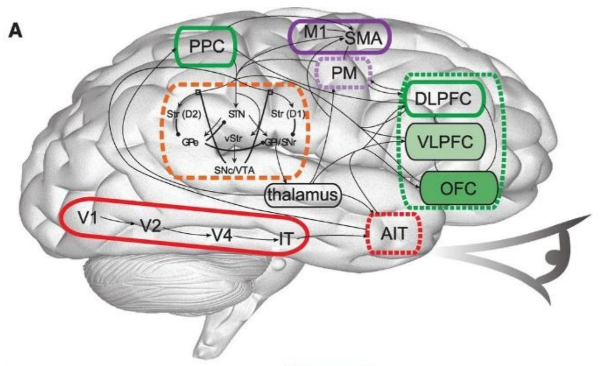




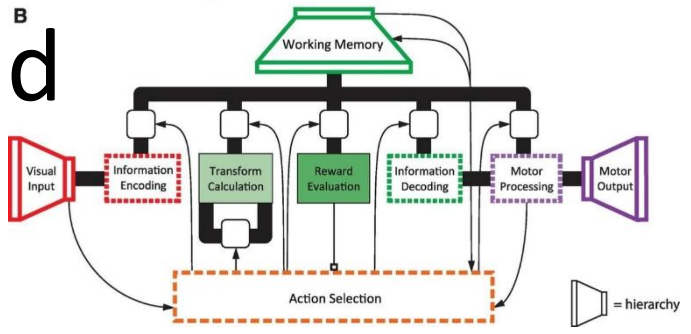


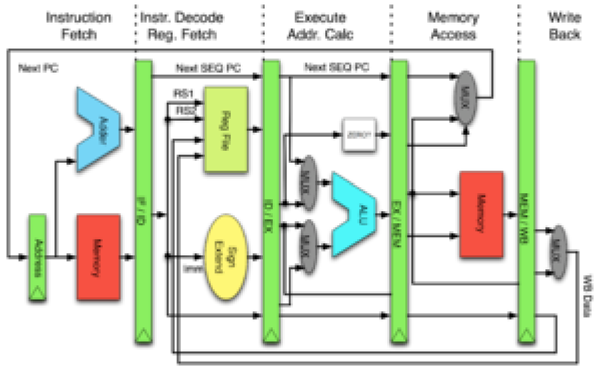
Architecture





- The process of planning, designing and constructing structures
  - Building
  - Business
  - Cognitive
  - Computer
  - Software
  - .....





What?
Who? & Where?
Why?
Why?
Stakeholders
Policies, Rules, Regulations
Capabilities
Vision, Strategies & Tactics
Who? & Where?
Organization
Information
What?
What?
Products & Services
Value Streams
Initiatives & Projects
How?
How?
Metrics & Measures
Decisions & Events
How Well?
When? & Where?
Business Architecture

Software Architecture

- Structures of a software system and the discipline of creating such structures and systems

Software Architecture

- Structures of a software system and the discipline of creating such structures and systems
- Each structure compromises
    - Software elements
    - Relations among the elements
    - Properties of elements
    - Properties of relations

Software Architecture

- Structures of a software system and the discipline of creating such structures and systems
- Each structure compromises
  - Software elements
  - Relations among the elements
  - Properties of elements
  - Properties of relations
- Blueprint for the system and the developing project, laying out the task necessary to be executed by the design teams

Software Architecture

- The software architecture of a program or computing system is the structure or structures of the system, which comprise software elements, the externally visible properties of those elements, and the relationships among them.

Service Oriented Architecture

- A style of software design where services are provided to the other companies by application components, through a communication protocol over a network

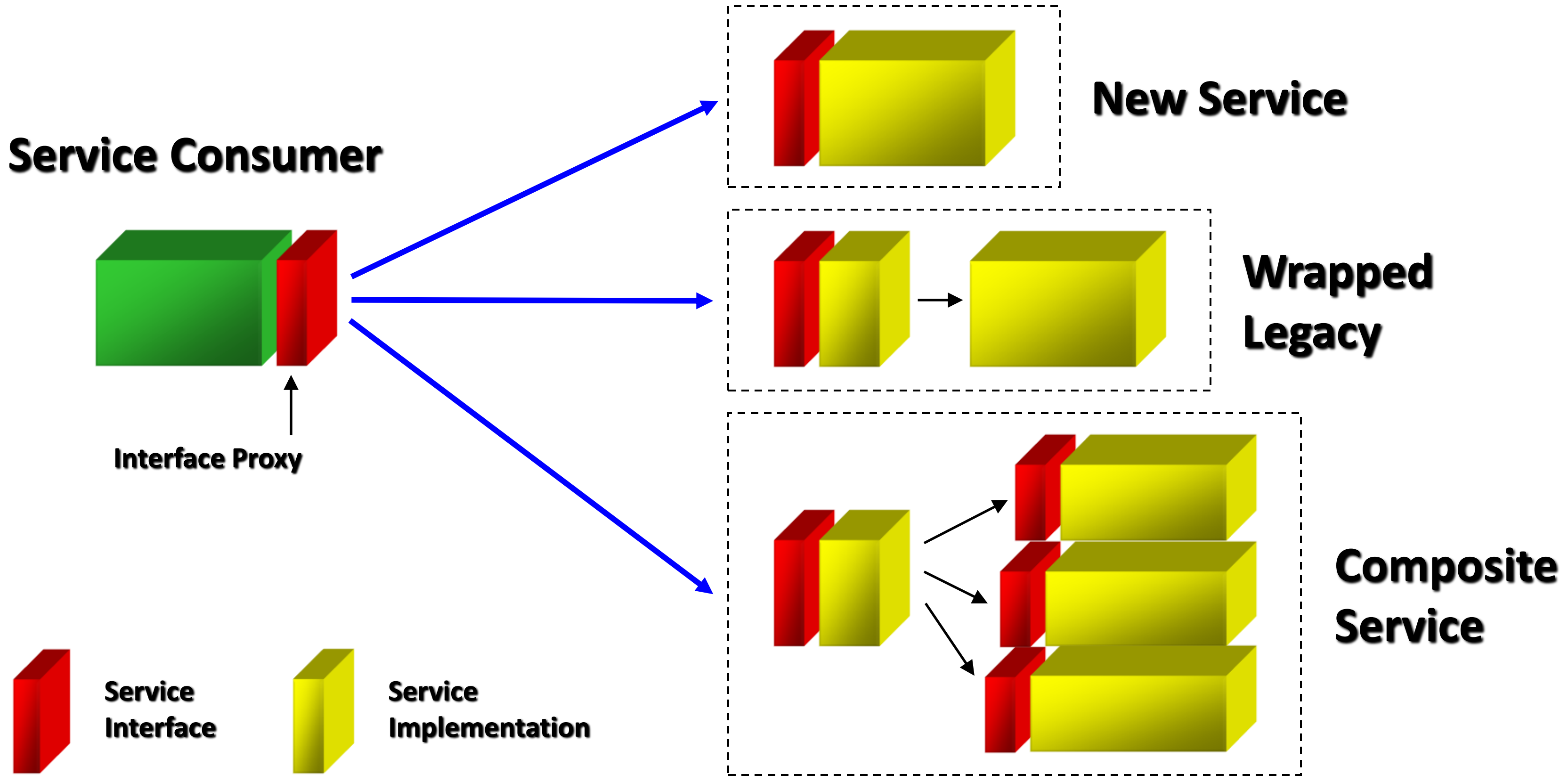
What is SOA?

- Service-Oriented Architecture (SOA) is a set of principles and methodologies for designing and developing software in the form of interoperable services.
- These services are well-defined business functionalities that are built as software components that can be reused for different purposes.
- SOA design principles are used during the phases of systems development and integration.

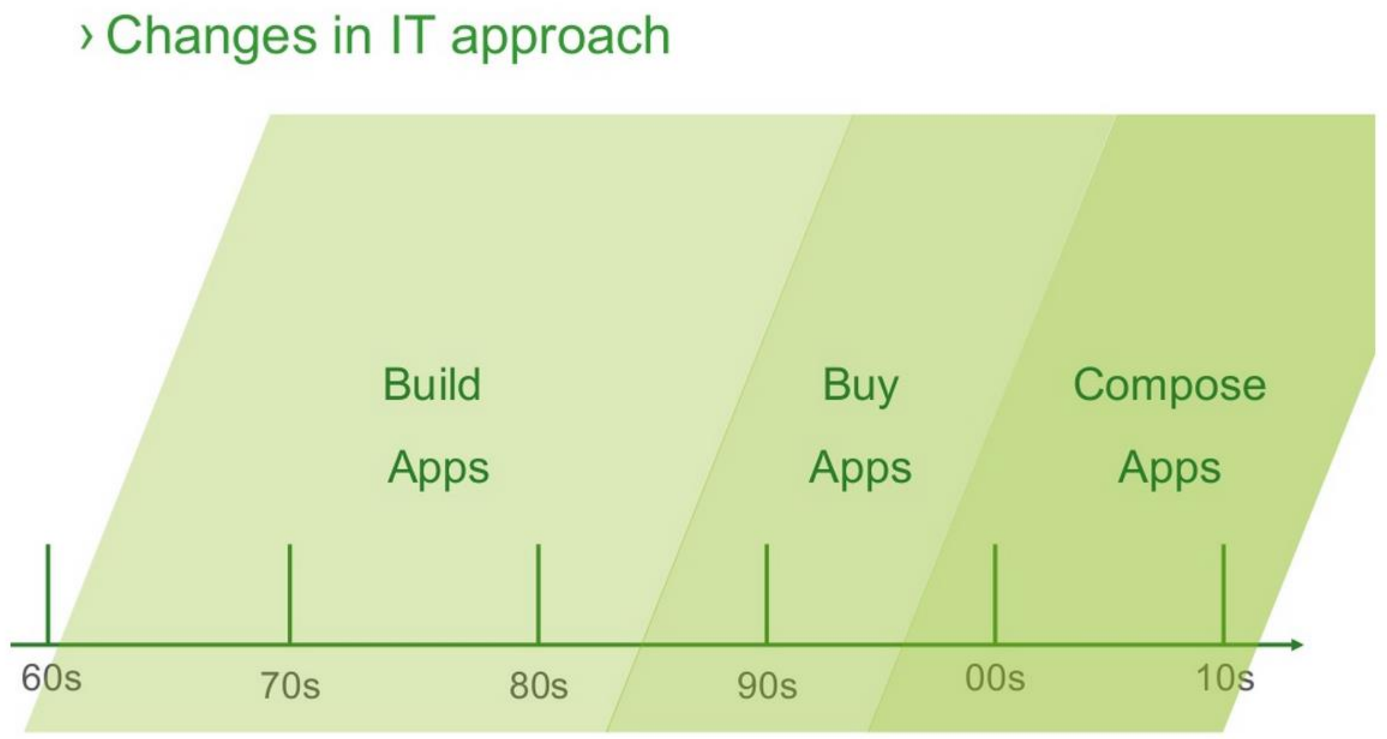
SOA Service

- A discrete unit of functionality that can be accessed remotely and acted upon and updated independently
- A repeatable task, for example:
    - Open an account
    - Perform a credit check
    - retrieving a credit card statement online

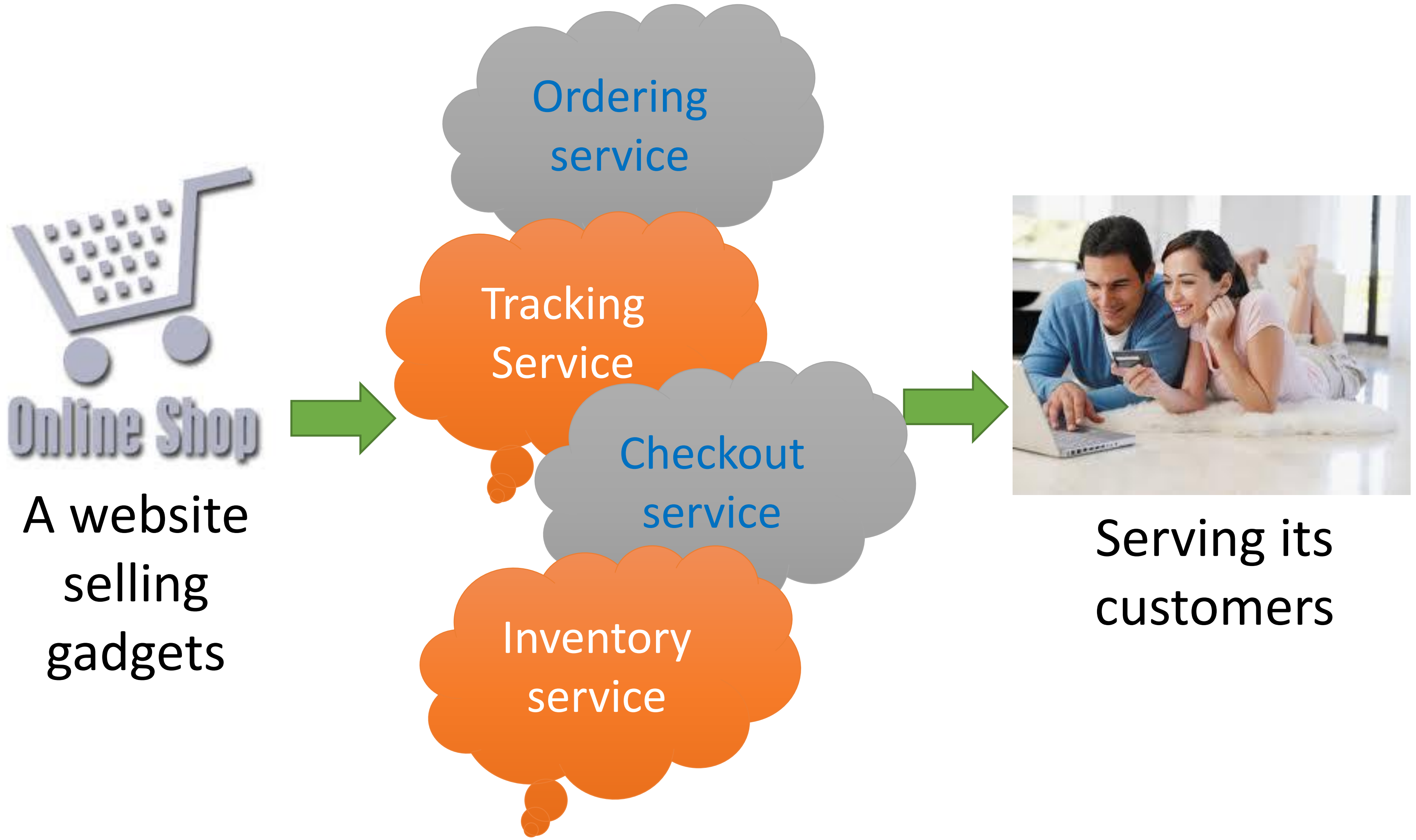
Anatomy of a Service



How did it come to SOA?



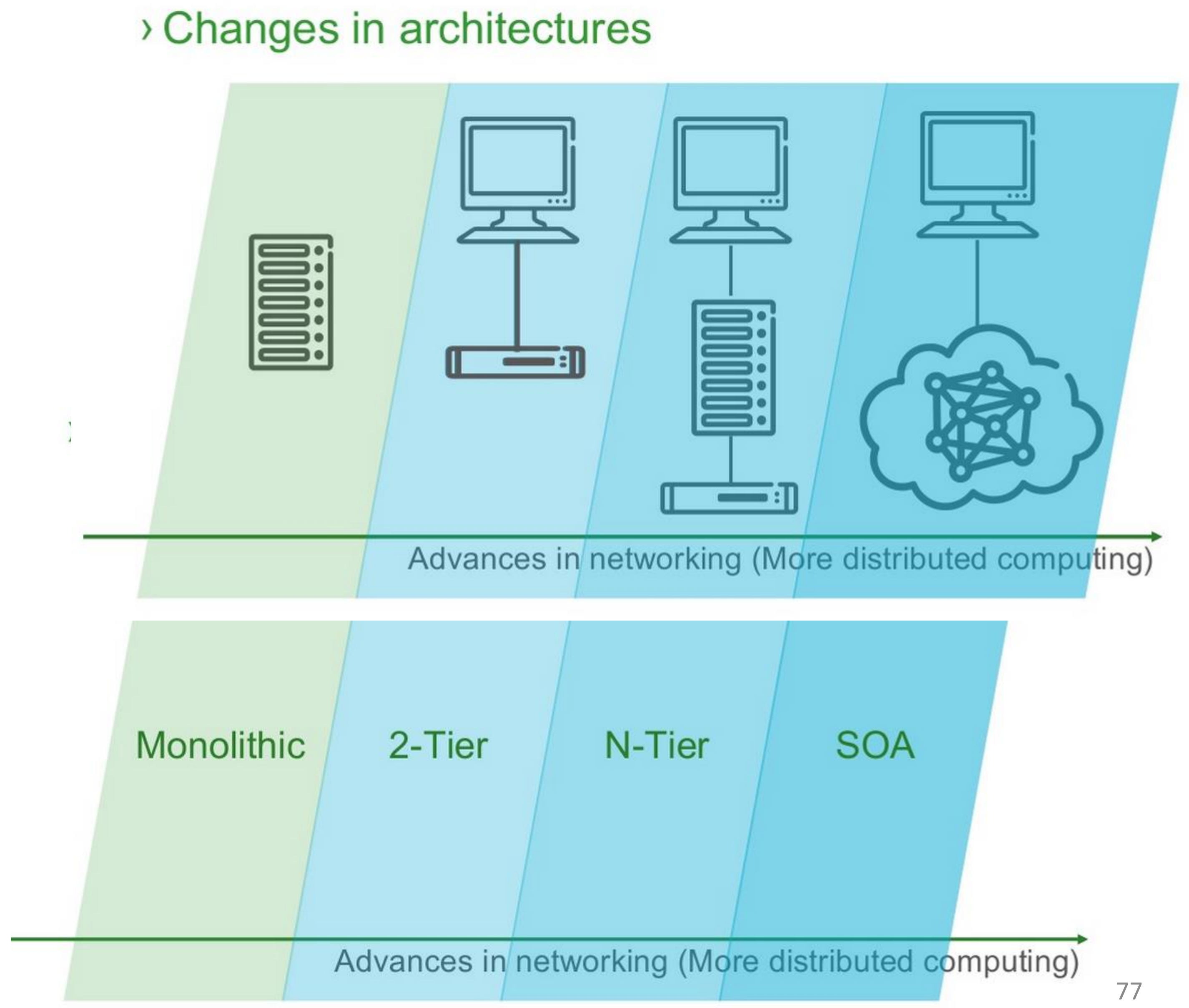
How did it come to SOA?



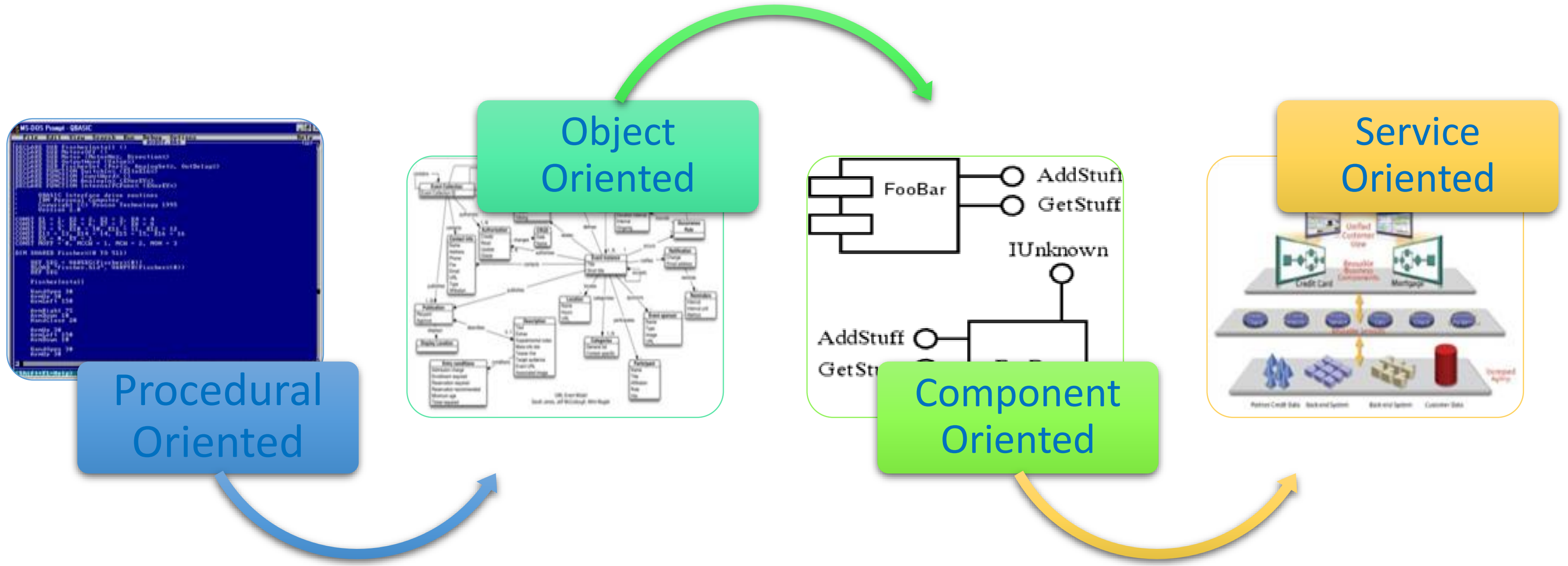
A website selling gadgets

Serving its customers

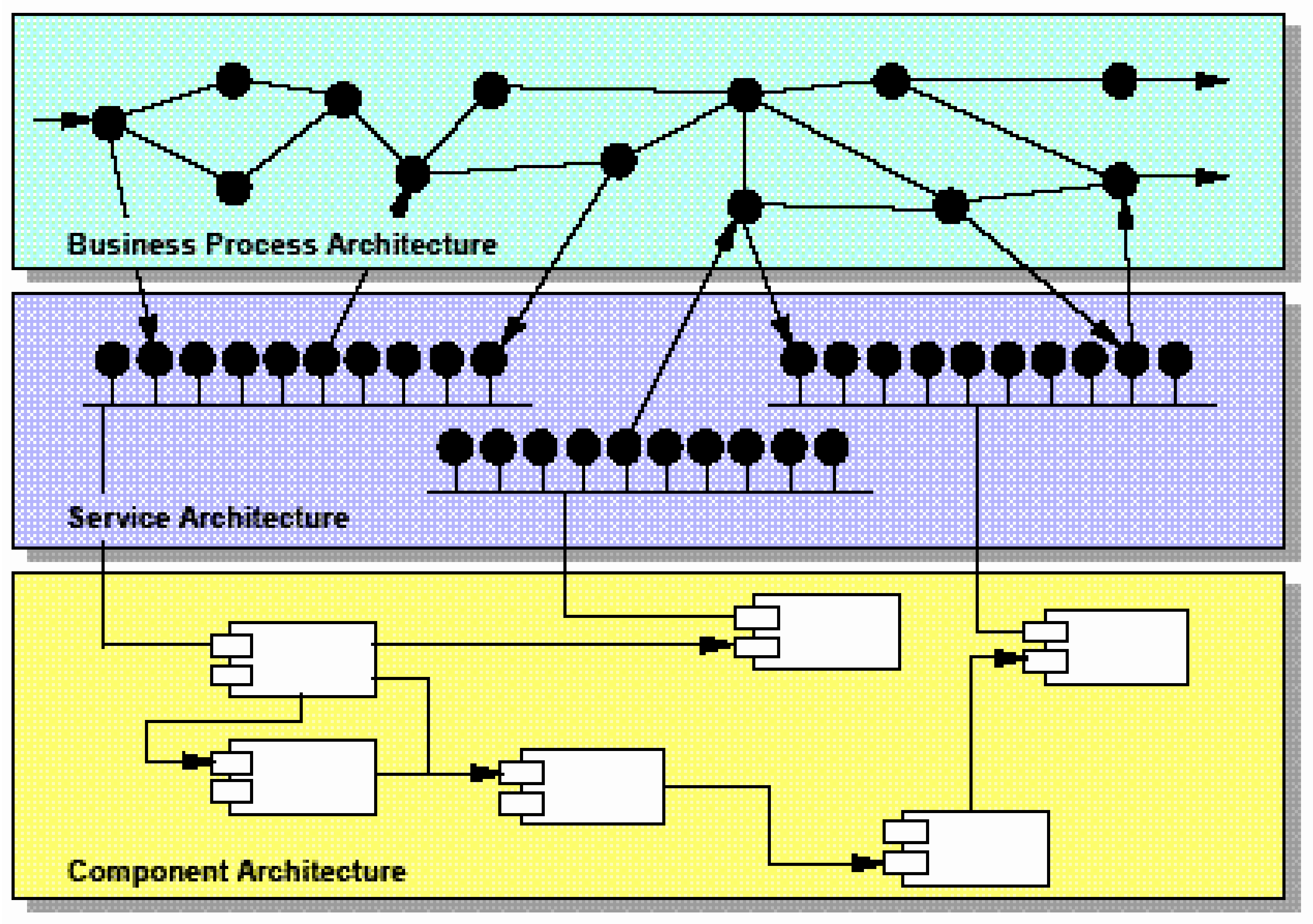
How did it come to SOA?


› Changes in architectures
Advances in networking (More distributed computing)
Monolithic
2-Tier
N-Tier
SOA
Advances in networking (More distributed computing)

How did it come to SOA?



Isolating the Business Process from the Implementation



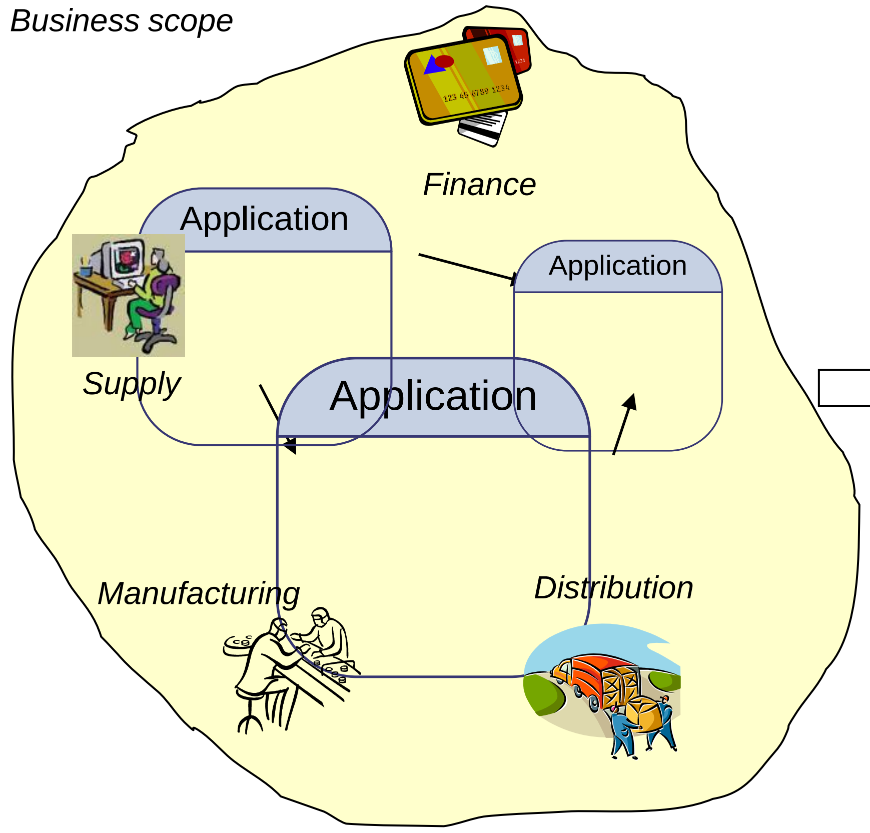
Why SOA in Enterprise?

- Why to introduce SOA in an organization? What are the benefits for enterprises?

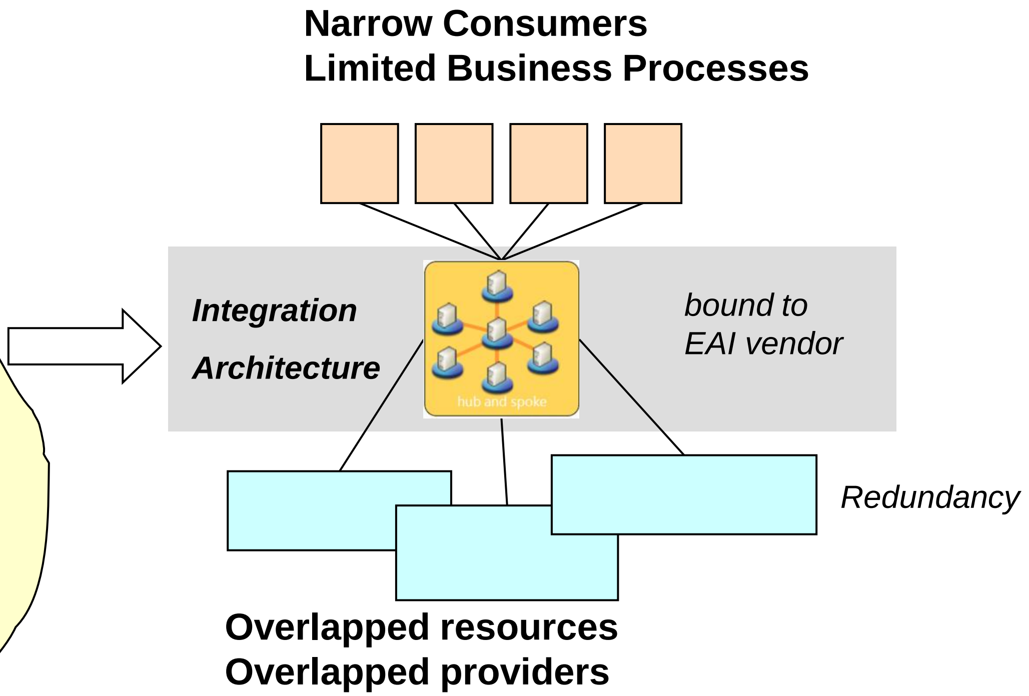
Why SOA?

- From application centric to service centric environment

Application Centric

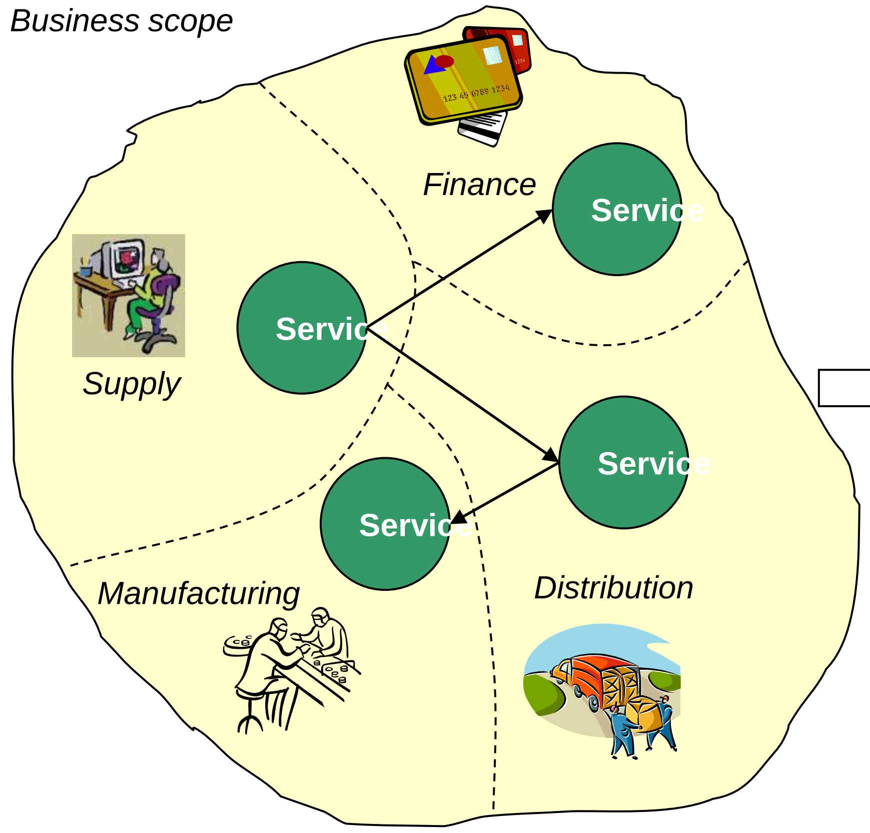


Business functionality is duplicated in each application that requires it.

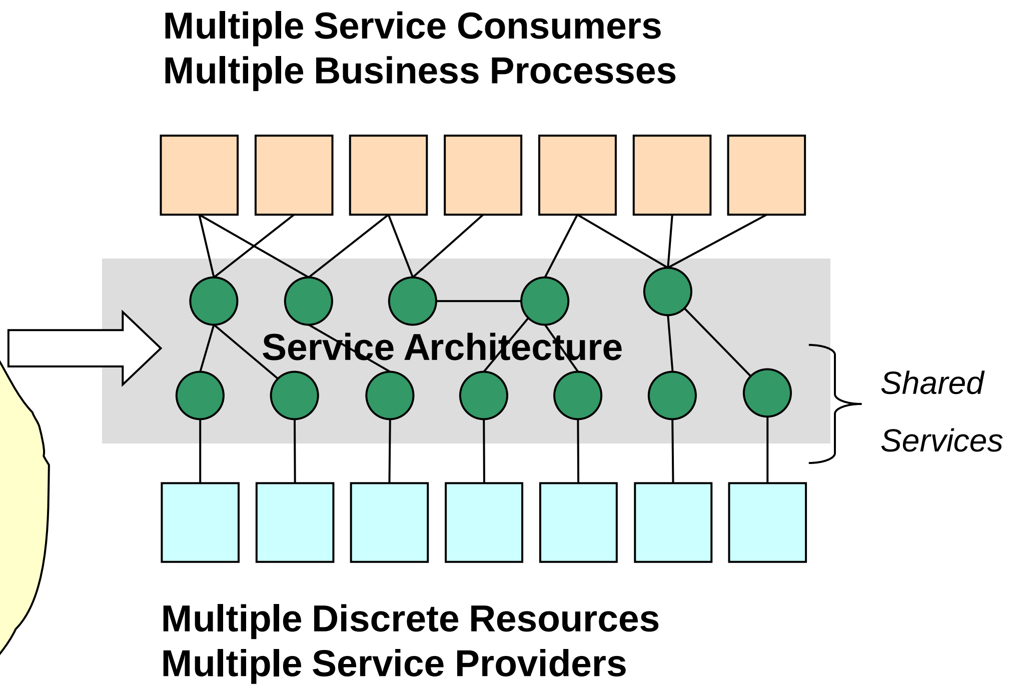


EAI ‘leverage’ application silos with the drawback of data and function redundancy.

Service Centric



SOA structures the business and its systems as a set of capabilities that are offered as Services, organized into a Service Architecture



Multiple Service Consumers
Multiple Business Processes
Service Architecture
Shared
Services
Multiple Discrete Resources
Multiple Service Providers

Service virtualizes how that capability is performed, and where and by whom the resources are provided, enabling multiple providers and consumers to participate together in shared business activities.

- We will continue this topic on Saturday