Architecture Paradigms for Robotic Applications

Monica Reggiani

Dept. of Management and Engineering
University of Padua

October, 2008

Introduction

- Robots supporting people in everyday tasks
  - deal with dynamic environments
  - ensure safe interaction with human beings
  - require complex multifunctional structure for control
  - interact with external systems (embedded computing and communication devices)
- Several architectures proposed:
  - orocos, marie, martha, miro, claraty, yarp, urbi, ...
  - lack of a common, suitable solution

Distributed Technology Research

- Distributed Object Architecture (DOA)
  - based on object oriented approach
  - improvement over first platform independent solutions (sockets, Java RMI, ...)
- Component Based Architecture (CBA)
  - based con the concept of software component
  - support of deployment of multiple components from multiple sources
- Service Oriented Architecture (SOA)
  - based on the concept of service
  - provide loosely coupled, highly dynamic applications

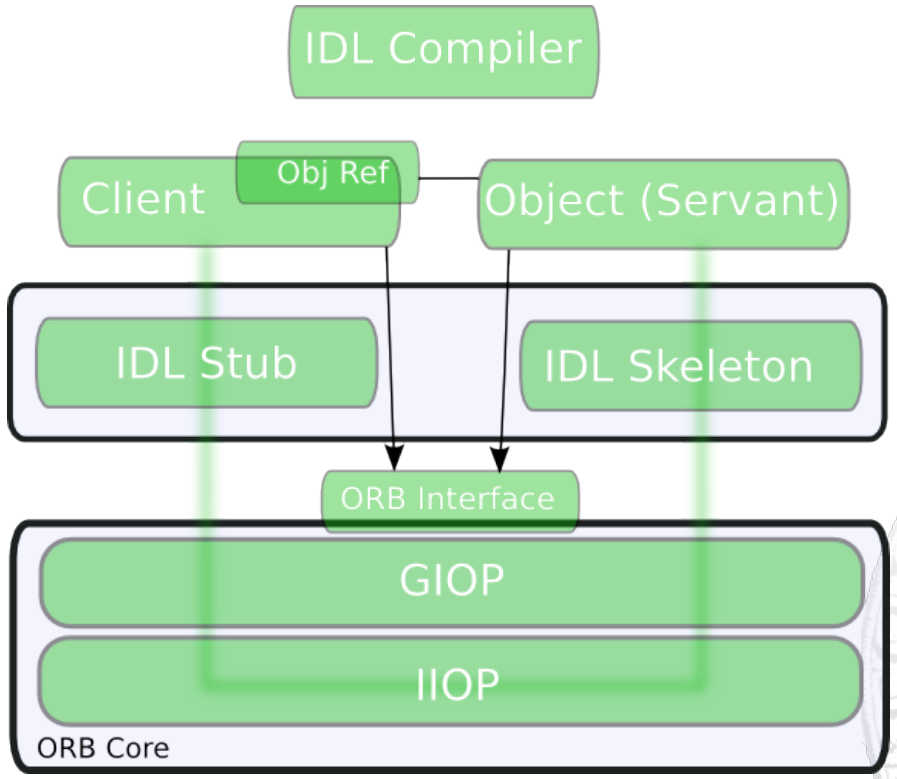
Distributed Object Architecture (DOA)

- DOA applications are:
  - *composed of objects, individual units of running software that combine functionality and data* (OMG)
  - run on multiple computers to act as a scalable computational resource
- Interaction supported through the definition of *interfaces*:
  - declare the available operations of a distributed object
  - clients know which requests they are allowed to perform
  - DOA system knows how to marshall/unmarshall the arguments
- Fine-grained interfaces, high level of control on concurrency

DOA Standards and Middlewares

Common Object Request Broker Architecture

- Common Object Request Broker Architecture (CORBA)
  - Vendor independent specification promoted by the Object Management Group (OMG)



DOA Robotic Applications

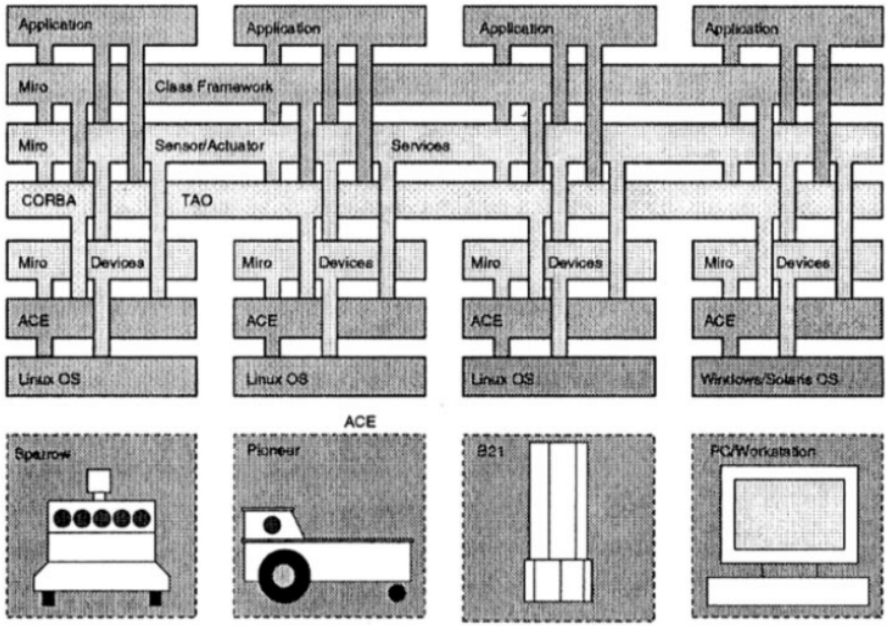
Early solutions

- Dynamic interconnection of heterogeneous and geographically distributed systems
  - Naming Service, Synchronous Method Invocation (SMI)
- Scalable and versatile data distribution tools
  - CORBA Event and Notification Service

DOA Robotic Applications

Early solutions

- Dynamic interconnection of heterogeneous and geographically distributed systems
- Scalable and versatile data distribution tools

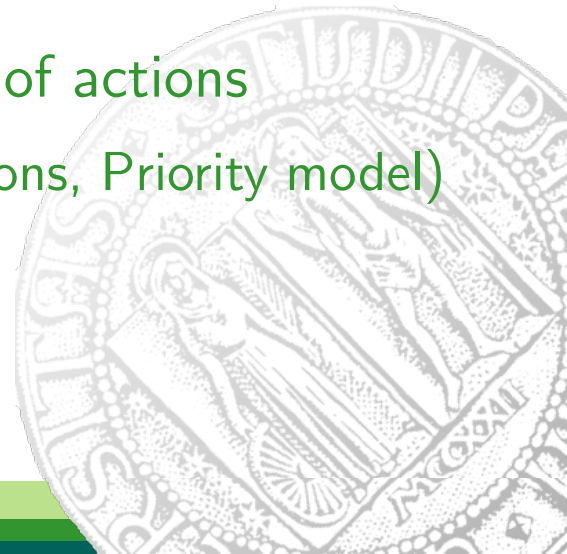


Source: *Miro - Middleware for Mobile Robot Applications*, H. Utz, S. Sablatnög, S. Enderle, G. Kraetzschmar

DOA Robotic Applications

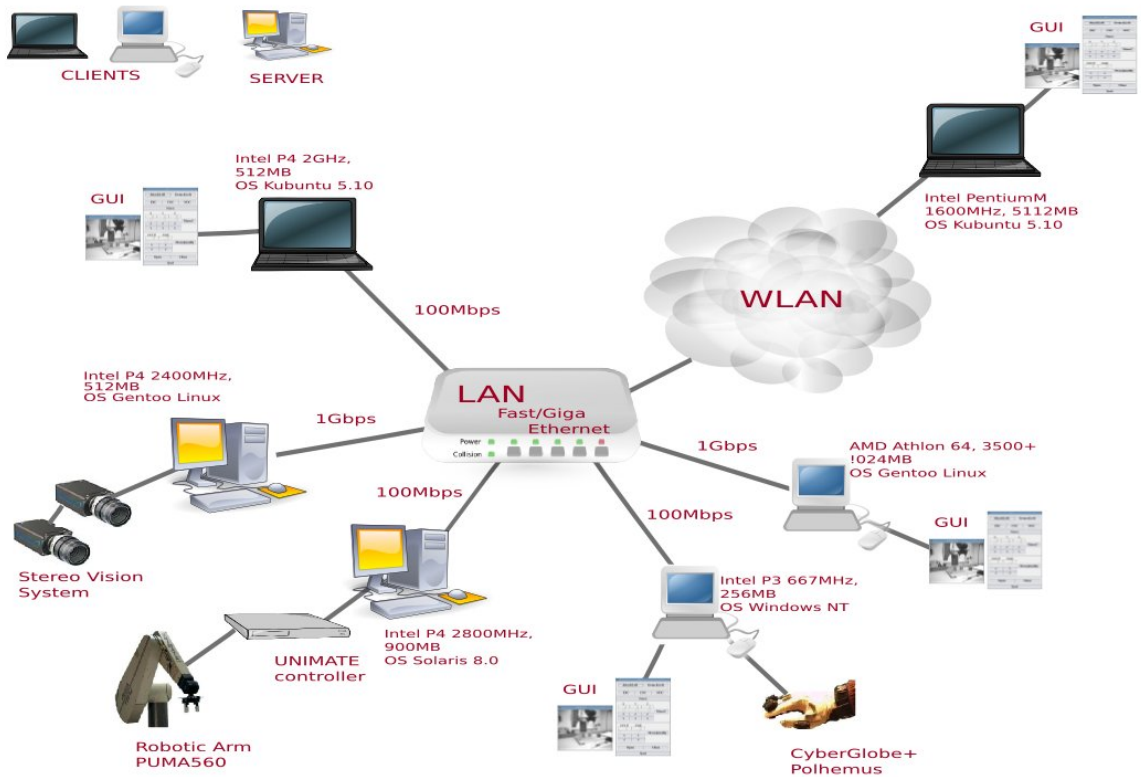
Real-time and Embedded Systems (1)

- Dynamic interconnection of heterogeneous and geographically distributed systems
  - Naming Service, Synchronous Method Invocation (SMI)
- Scalable and versatile data distribution tools
  - CORBA Event and Notification Service
- Concurrent execution of several tasks
  - Asynchronous Method Invocation (AMI)
- Real-Time requirements and control over priority of actions
  - Real-Time CORBA (Threadpool, Banded Connections, Priority model)
- Security and concurrency management
  - CORBA Concurrency and Security Service



DOA Robotic Applications

Real-time and Embedded Systems (2)



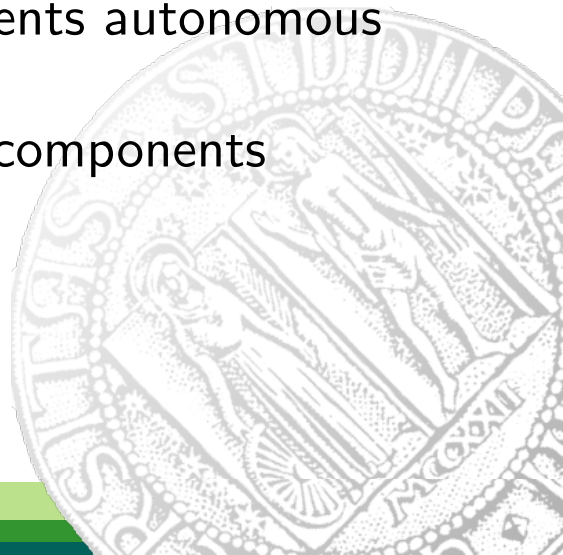
Source: *Designing Distributed, Component-based Systems for Industrial Robotic Applications*,

M. Amoretti, S. Caselli, M. Reggiani

Component Based Architecture (CBA)

Introduction

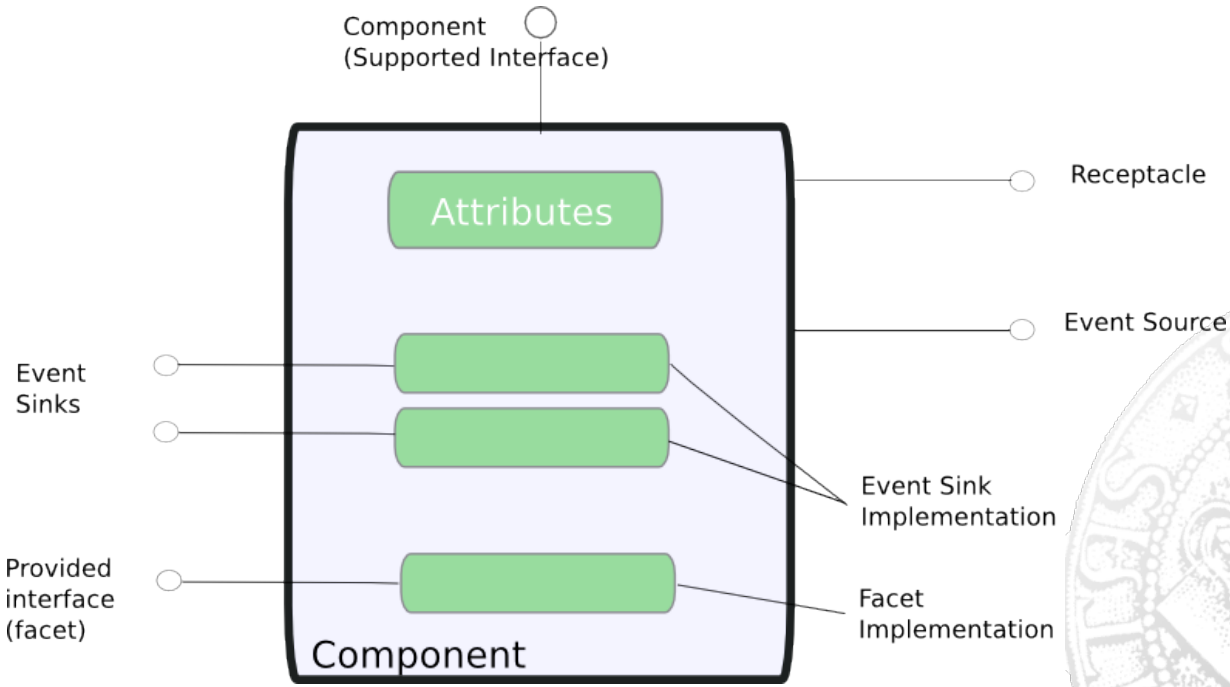
- Software component:
  - unit of composition with contractually specified interface
  - strong separation between interface and implementation
- Objects are not candidate for components:
  - objects exist at run time, components are binaries that are deployed
  - objects tightly coupled with other objects, components autonomous units whose purpose is well defined and understood
  - objects are generally much more fine-grained than components



CBA Standards and Middleware

CORBA Component Model (CCM)

- CORBA Component Model (CCM) introduces:
  - features to simplify and automate the construction, composition, and configuration of components
  - steps in the application development lifecycle



CBA Standards and Middleware

Internet Communication Engine (ICE)

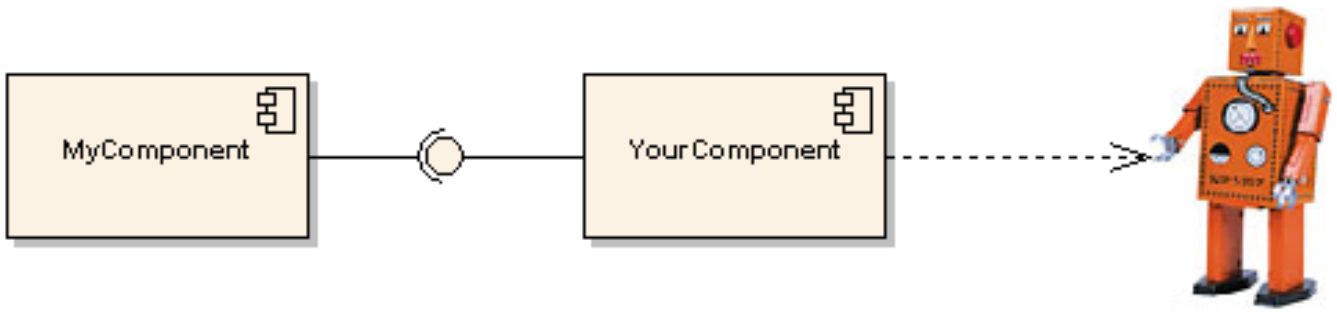
- Developed by ZeroC group as an alternative to CORBA OMG standard
- Aims at avoiding unnecessary complexity
- Supports a large number of languages (C/C++, Java, C, PHP, Visual Basic)
- Two main services:
  - ICEGrid
  - ICEStorm

www.zeroc.com/ice.html

CBA Robotic Applications

Why are components a good idea for robotics?

- Effort required to develop complete control software before being able to start with the implementation of research issues:
  - develop components for mature algorihtms, sensors, and actuators
- Domain characteristics particularly suited the CBA approach:
  - inherent complexity, requirement for flexibility, distributed environments, heterogeneity of hardware and operating systems.

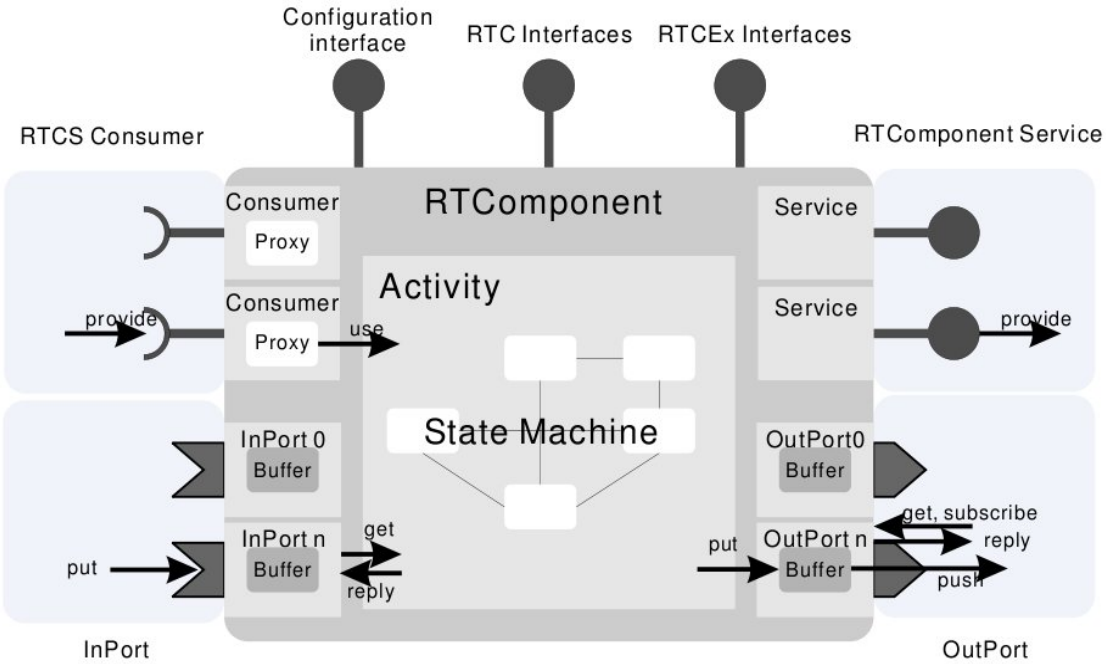


Source: *Towards component-based robotics*, A. Brooks et al.

CBA Robotic Applications

RT-Middleware

- simplify system integration through a methodology for the creation of robotics technology component and a framework for their composition



Source: *RT-Middleware: Distributed Component Middleware for RT (Robotic Technology)*. N. Ando et al.

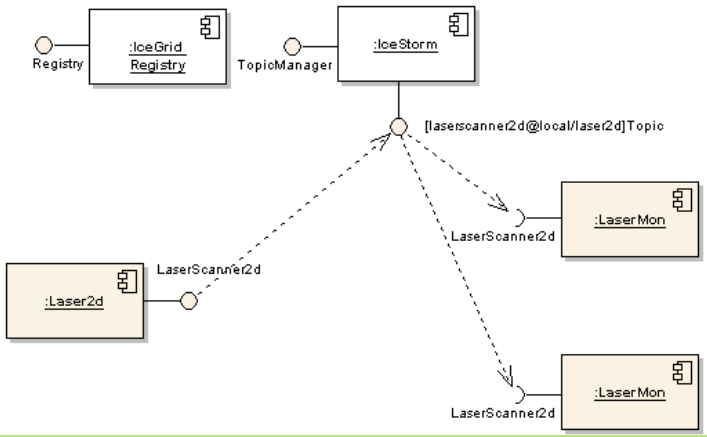
CBA Robotic Applications

ORCA

- Open source implemntation framework for developing component based robotic systems.

Source:`www.zeroc.com/ice.html`

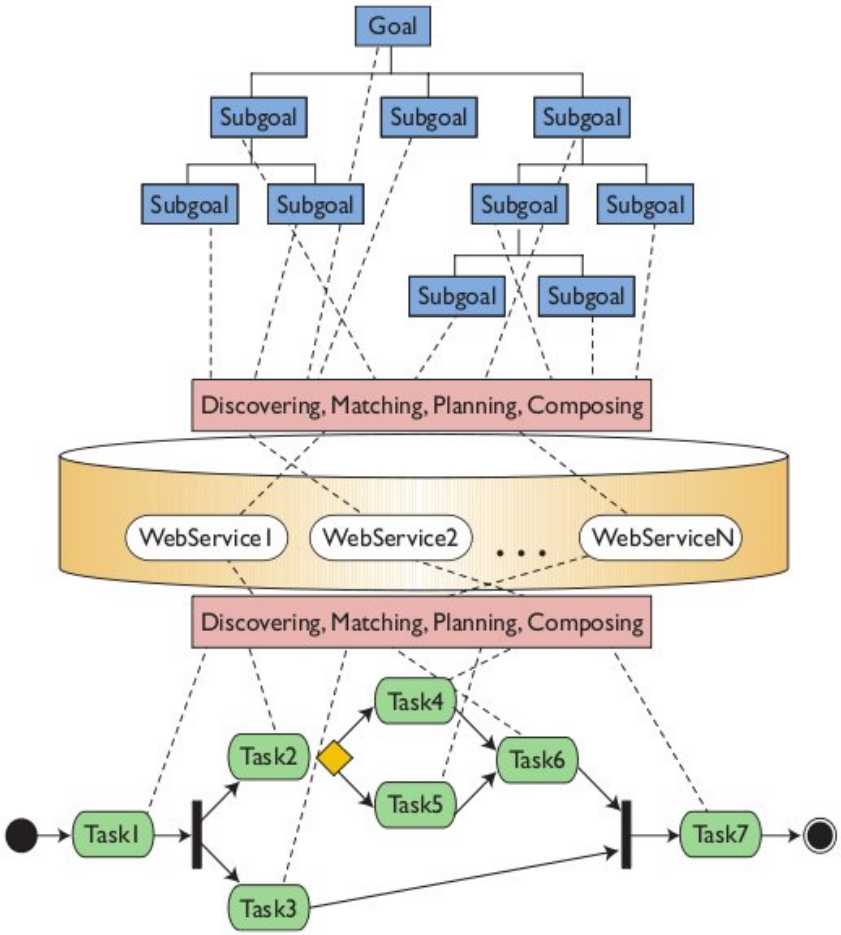
- Elements of the Orca framework:
  - Objects
  - Communication Patterns
  - Transport Mechanism
  - Components



Service Oriented Architecture

- based on the concept of service,
  - a unit of work executed by a service provider to achieve the desired results for a service consumer
  - higher-level abstraction for organizing applications for large scale, open environment.
- Key elements for SOA:
  - Loose coupling
  - Implementation neutrality
  - Flexible configurability
  - Persistence
  - Granularity
  - Teams

Service Oriented Architecture



Source: *Service-Oriented Computing: Key Concepts and Principles* M. N.Huhns, M. P. Singh

Service-Oriented Architecture

OWL-S

- OWL-S, an ontology built on top of Web Ontology Language (OWL) for the Semantic Web .
- Objectives:
  - automatic service discovery
  - automatic service invocation
  - automatic service composition and interoperation
- ... still under development

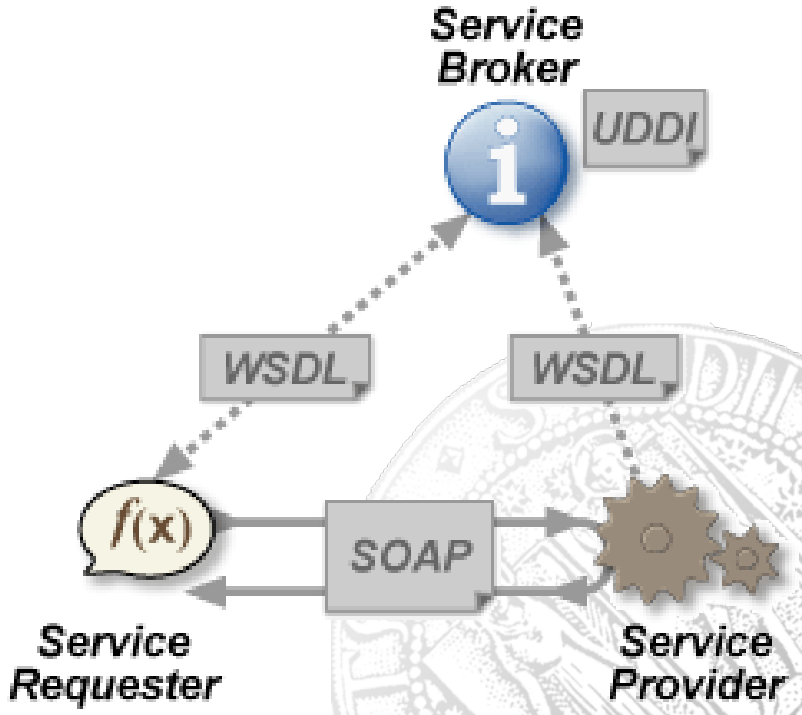
Service-Oriented Architecture

Web Services

- Web service: "a software system designed to support interoperable machine-to-machine interaction over a network" [W3C]

Protocols and standards:

- Web Services Description Language (WSDL)
- Universal Description, Discovery and Integration (UDDI)
- SOAP
- XML



Source: en.wikipedia.org/wiki/Web_service

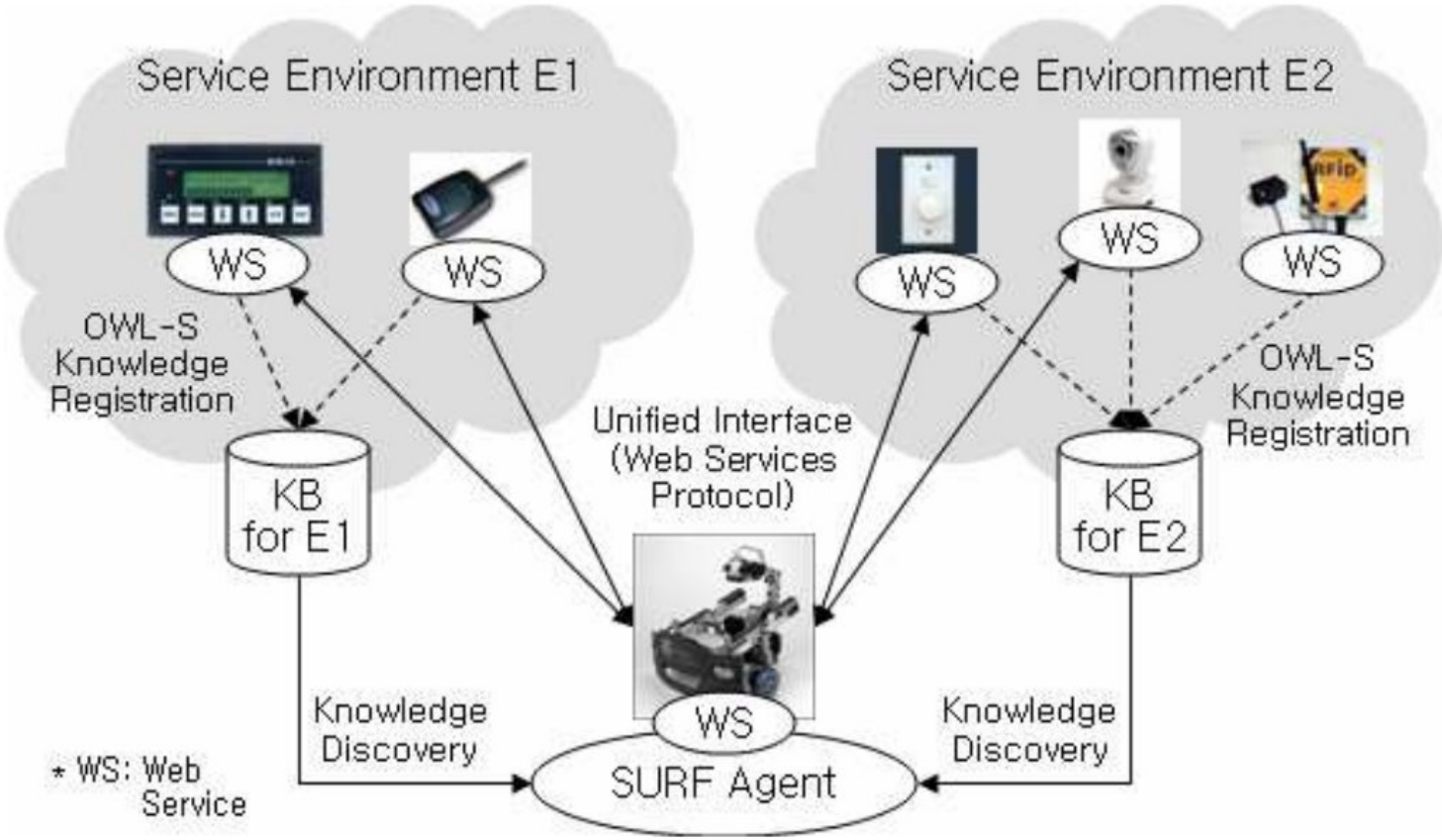
SOA Robotic Applications

- First phase:
  - limited exploitation of SOA protocols and tools
- Second phase:
  - (re)design according to service-centric models (OWL-S, WSRF)

SOA Robotic Applications

Ubiquitous **R**obotic **S**ervice **F**ramework (1)

- Traditional networked robotic system

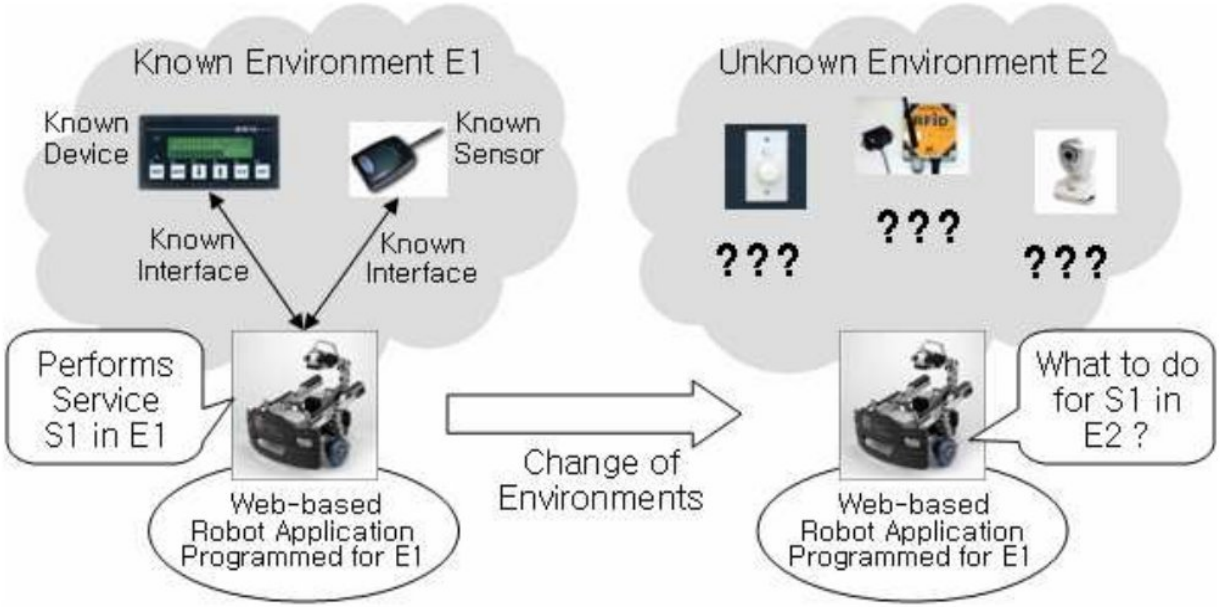


Source: *Service-Oriented Integration of Networked Robots with Ubiquitous Sensors and Devices Using the Semantic Web Services Technology*, Young-Guk Ha, Joo-Chan Sohn and Young-Jo Cho

SOA Robotic Applications

Ubiquitous **R**obotic **S**ervice **F**ramework (2)

- USRF approach

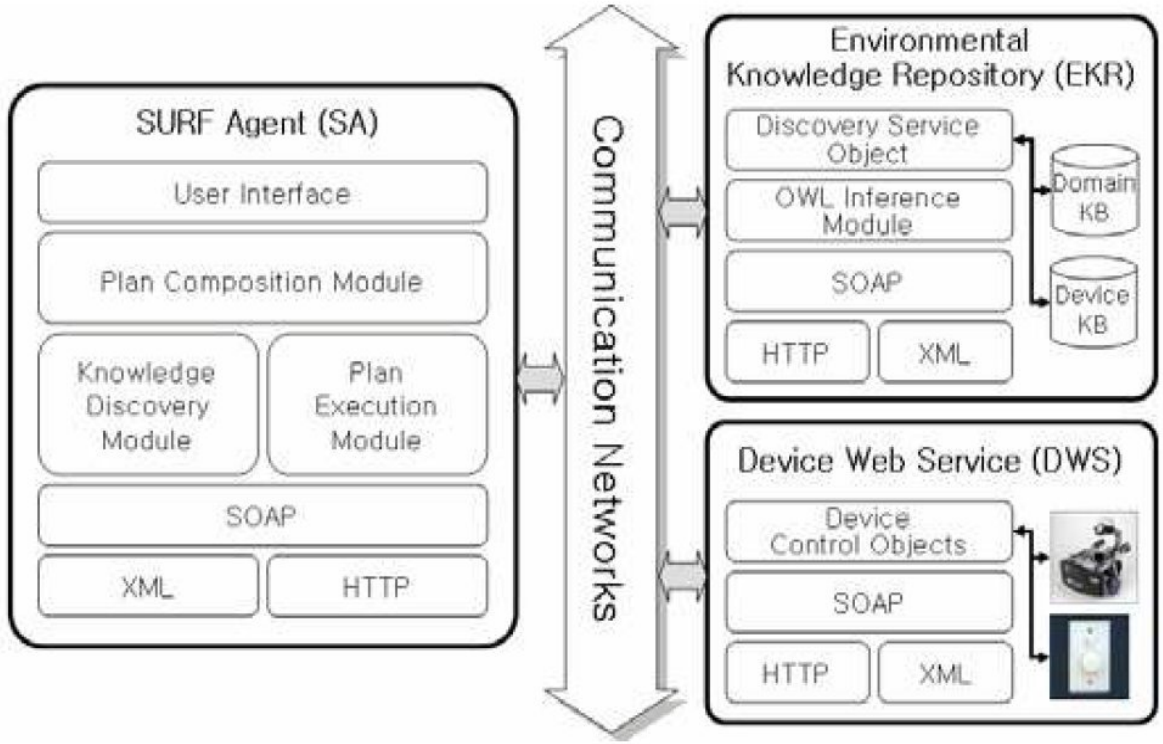


Source: *Service-Oriented Integration of Networked Robots with Ubiquitous Sensors and Devices Using the Semantic Web Services Technology*, Young-Guk Ha, Joo-Chan Sohn and Young-Jo Cho

SOA Robotic Applications

Ubiquitous **R**obotic **S**ervice **F**ramework (3)

- Three main components: *Robotic Agent*, *Environmental Knowledge Repository*, *Device Web Services*

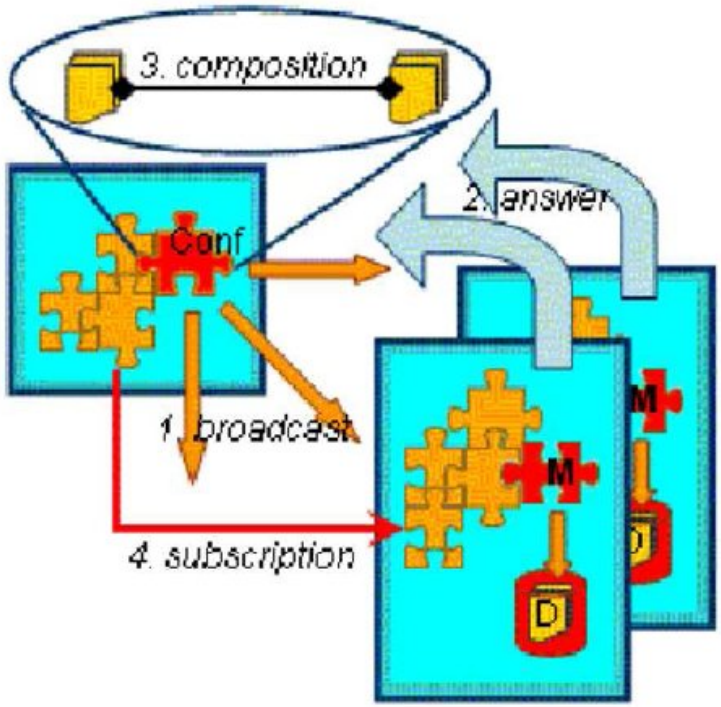


Source: *Service-Oriented Integration of Networked Robots with Ubiquitous Sensors and Devices Using the Semantic Web Services Technology*, Young-Guk Ha, Joo-Chan Sohn and Young-Jo Cho

SOA Robotic Applications

PEIS Ecology

- formal description of the functionalities;
- framework for discovery and run-time composition;
- mechanism for semantic interoperability



Source: *Reactive Self-Configuration of an Ecology of Robots*, Marco Gritti, Mathias Broxvall, Alessandro Saffiotti

Conclusions

- Different paradigms have different characteristics and properties making them suitable for different robotic applications.

- Distributed Object Architecture (DOA)
  - fine-grained concept of **object**, suitable for lower layers;
- Component Based Architecture (CBA)
  - suitable for mid-tiers to develop autonomous **components** that can be exchanged and composed;
- Service Oriented Architecture (SOA)
  - loosely couple architecture where interacting entities (**service**) can be accessed without previous knowledge