Semantic Interoperability in Service-Oriented Computing

Giovanna Di Marzo Serugendo

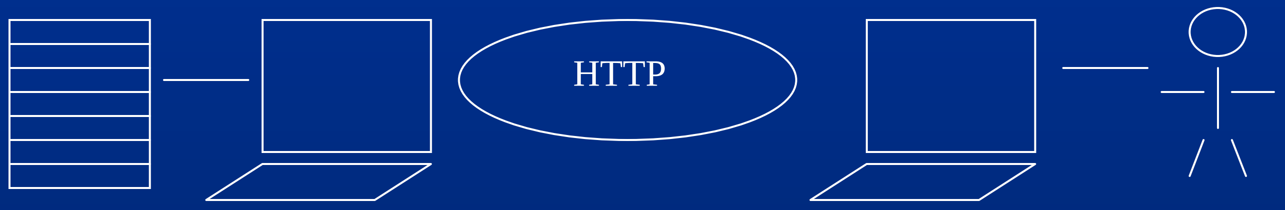
University of Geneva, Switzerland

Outline

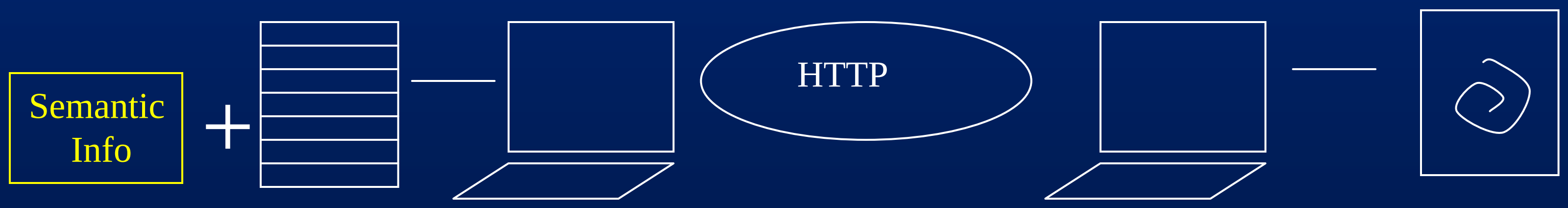
- Service-Oriented Computing
- Interoperability
- Syntactic Understanding
- Semantic Understanding
  - Ontology
  - Meta-ontology
- Self-Managing Systems

Service-Oriented Computing (SOC)

Traditional Web

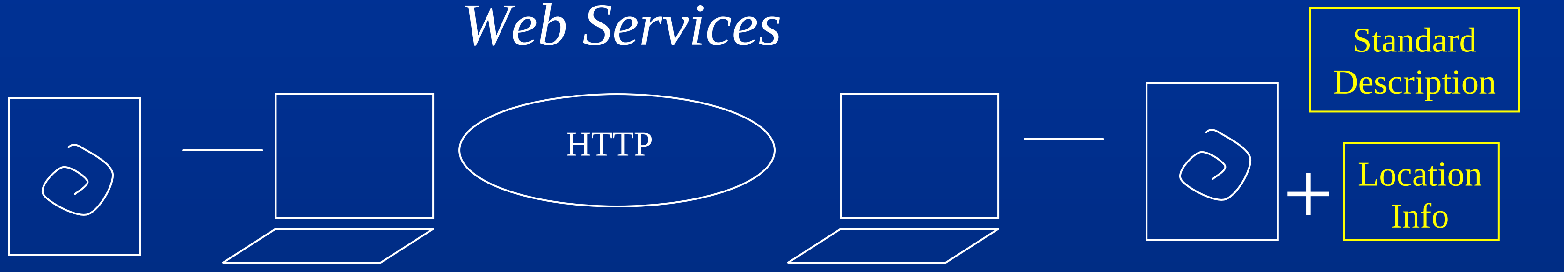


Semantic Web



Service-Oriented Computing (SOC)

Web Services



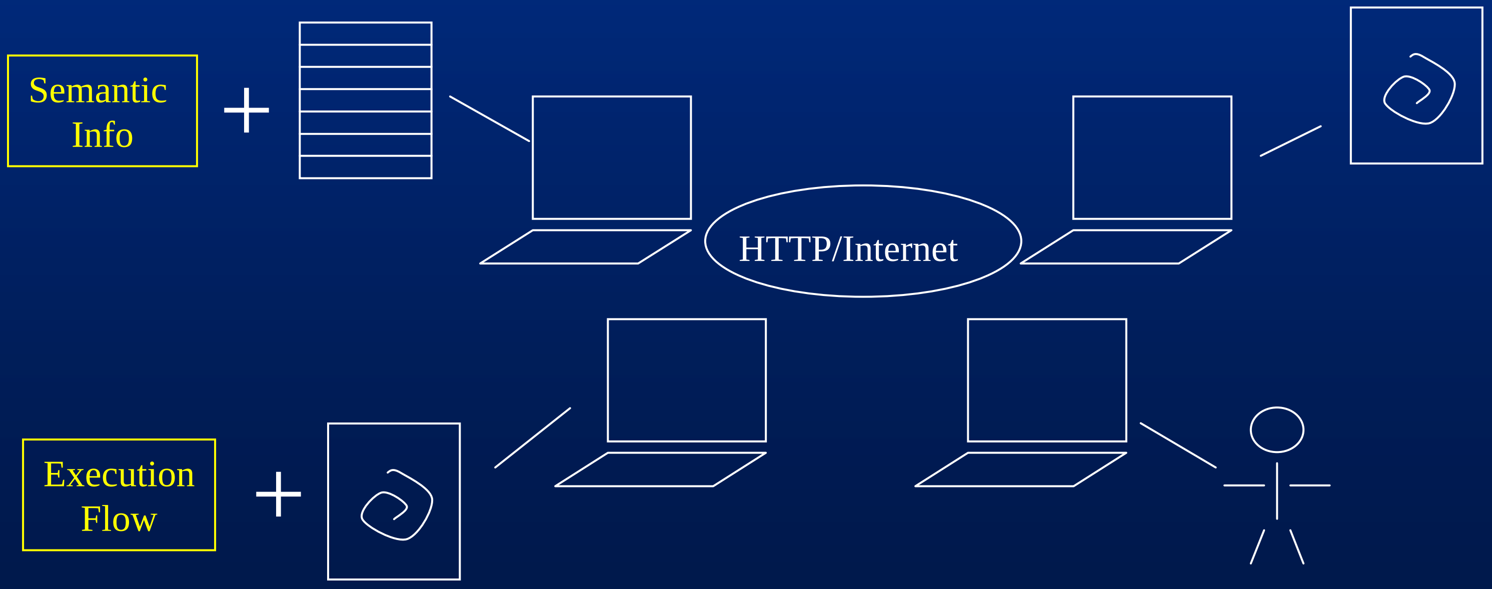
HTTP

Standard Description

+

Location Info

Service-Oriented Computing



Semantic Info

+

HTTP/Internet

Execution Flow

+

Service-Oriented Computing (SOC)

- Characteristics
  - Service = capability provided and exploited (not always) remotely
  - Autonomous Entities
    - Independence of:
      users / designers / administrators
  - Active Entities
    - Processes / Users
    - Take decisions / initiatives
  - Heterogeneous Entities
  - Collaboration among entities
  - Dynamic (join/leave)

- Interest
  - Semantically described services
  - Long-lived interactions
  - Negotiations
  - Interoperability
  - Grid Computing
  - Autonomic Computing

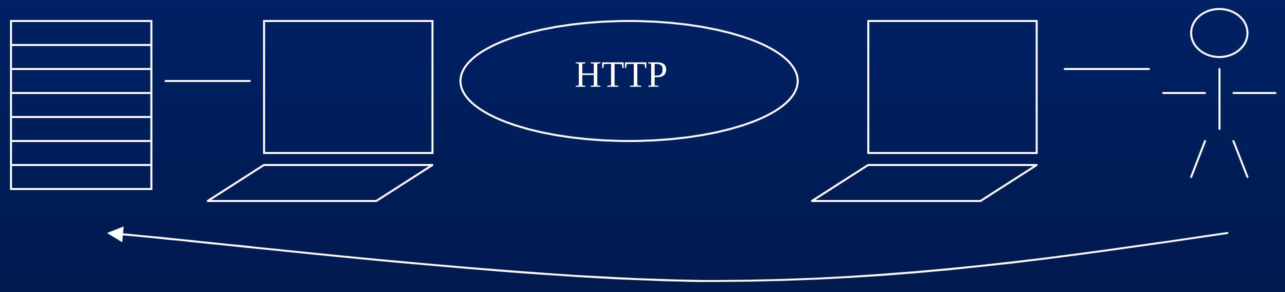
Interoperability

- Intra-enterprise interoperability
  - Different (initially independent) software need to work together
  - Issues: need for connectivity, mutual understanding, communications
- Inter-enterprise interoperability
  - Communication and understanding of information
  - Issues: need for agreement on data format
- SOC good for:
  - Building processes over systems
  - Local autonomous policies with coherent cross-enterprise processes
- Issues
  - Exchange information successfully
  - Semantics associated with information
  - Execution flows (processes)

Syntactic Interoperability

- Hard-coding of interoperability
  - No adaptability
  - No negotiation
- Same data structure / same API
  - Exact knowledge of method call
  - Glued together: data - application logic

Traditional Web

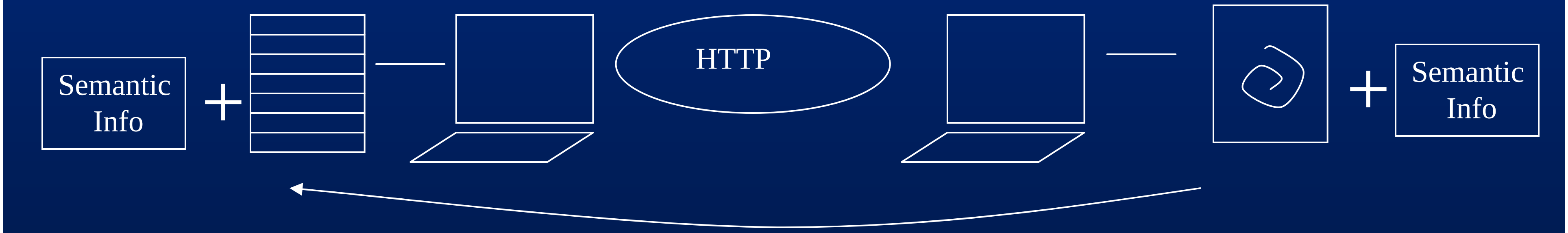


http://www.unige.ch

Semantic Interoperability

- Decoupling:
  - Data – application logic
    - Common data exchange format

Semantic Web

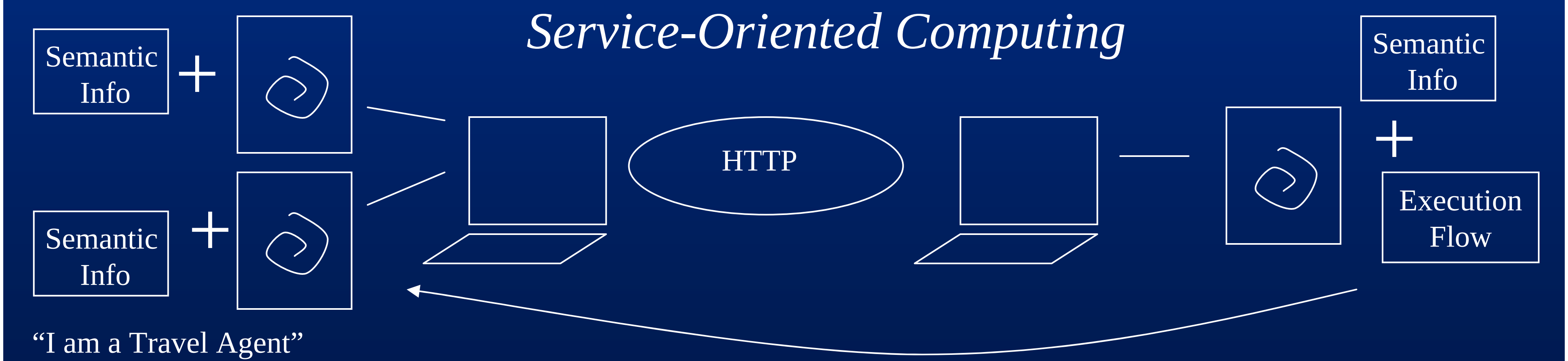


"What is the weather forecast in Geneva for the next two days?"

Semantic Interoperability

- Decoupling:
  - Data – application logic – control flow
    - Common data format + handling of dependencies

“I am hotel in Geneva”



Service-Oriented Computing

Semantic Info +

Semantic Info +

HTTP

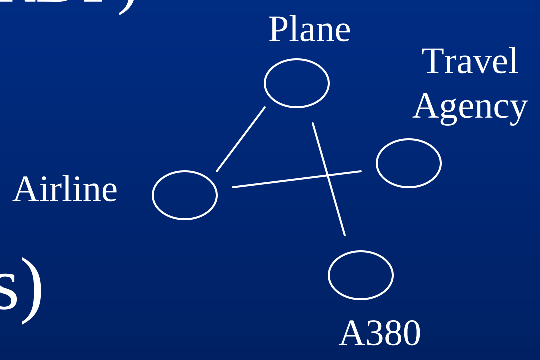
Semantic Info + Execution Flow

“I am a Travel Agent”

“I want to book a plane for Geneva (1-2/06/05), and then a hotel for the duration of my stay”

Semantic Interoperability - Tools

- Ontology
  - Knowledge representation describing a conceptualisation of some domain (RDF)
    - Vocabulary (keywords)
    - Semantic interconnection
    - Rules of inference



- Meta-Ontology (logical languages)
  - Go beyond ontology agreements
    - Constraints / new concepts (OWL)
  - Specification-Carrying Code (SCC)

SCC - Interest

- Decoupling of:
  - Application Logic (code)
  - Description of Code Functional behaviour
  - Description of Code Non-functional properties
  - Execution Flow
- Minimum basis for communication
  - Specification language (for expressing concepts)
- Interoperability with unknown software
  - No common design / No common API
- Seamless Integration of new entities
- Robustness

SCC for Self-Managing Systems

- Self-Configuration (installation, configuration, integration)
  - SCC expresses configuration policies (high-level, local)
    - Unanticipated dynamic run-time evolution of code
      - Seamless integration of new components

- Self-Optimisation (parameters)
  - SCC expresses optimisation policies (description / optimisation of parameters)
    - SCC Middleware seeks optimised service (most recent, most efficient, etc.)

- Self-Healing (error detection, diagnostic, repair)
  - Generation of correct code / Replacement of error code / Checking of code against specification

- Self-Protection (detection and response to attacks)
  - SCC expresses security policies (contracts, signatures of attacks, response schema)

Conclusion

- “Understanding” is fundamental
  - Data
  - Processes
  - Execution flows
- Service-Oriented Computing
  - Complex applications
  - Autonomy / heterogeneity
  - Interoperability
  - Negotiations

References

- « Service-oriented computing ». Munindar Singh, Michael Huhns. Wiley. 2005.