ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA
SEDE DI CESENA

aliCE

WOA 2010

Developing Web *Client* Applications with JaCa-Web

Mattia Minotti, Andrea Santi, Alessandro Ricci

DEIS, Università di Bologna

Cesena (FC) Italy

BACKGROUND OBJECTIVE

- Investigating Agents and MAS as a ***paradigm*** for computer programming and software development
  - *programming language* perspective, in particular
    - (multi-)agent programming languages

- **Agent-Oriented Programming** as a **post-OO** computer programming paradigm
  - supporting a **decentralized mindset** in problem solving, designing systems, programming
  - providing an effective **level of abstraction** for defining both
    - structure and behaviour of autonomous entities, and...
    - ...interaction and coordination among such entities

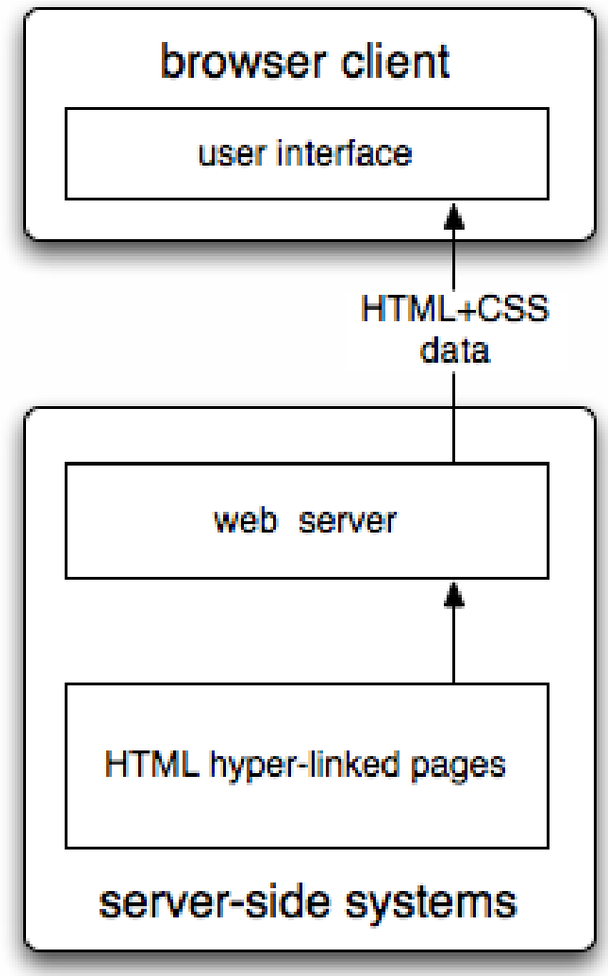
CURRENT INVESTIGATION

- Two main directions
  - stressing existing agent programming technologies
    - **JaCa** platform
  - exploring a family of new agent post-OO programming languages
    - **simpAL** ongoing project

- Evaluation
  - applying the approach to the development of hot real-world application contexts
    - desktop app
    - web app (=> **JaCa-Web**)
    - nomadic app (=> **JaCa-Android**)
    - ...

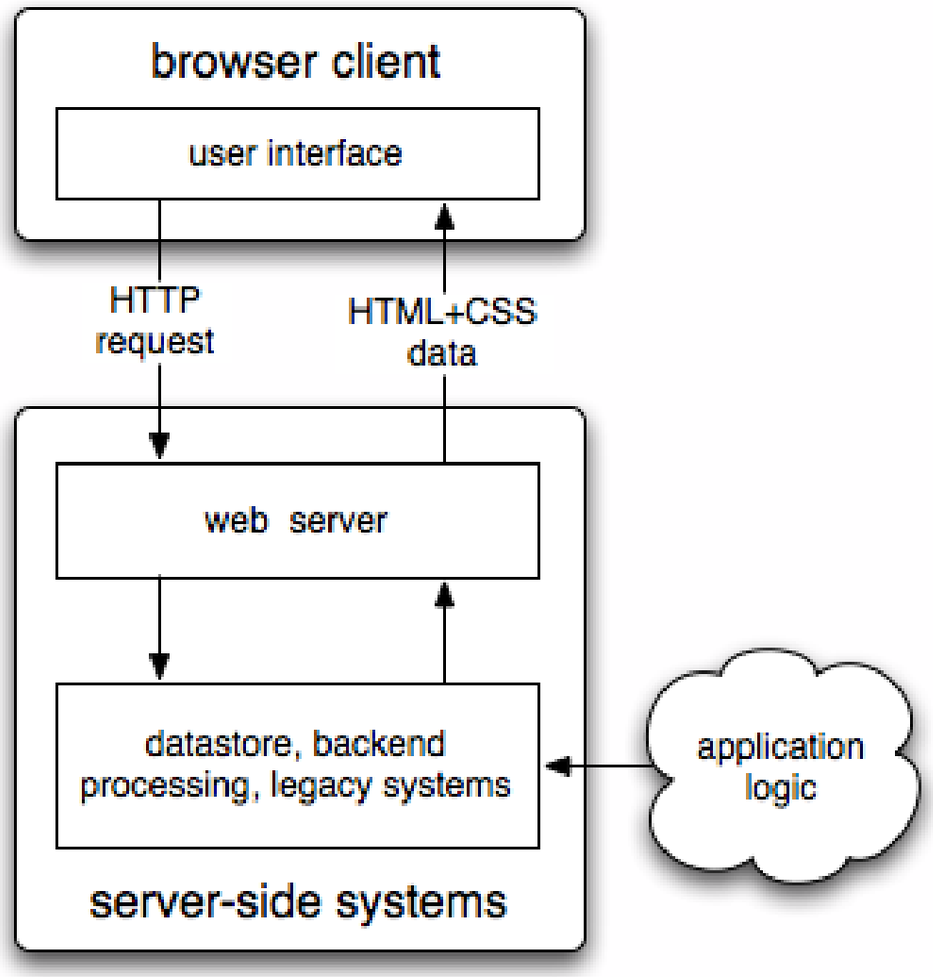
CLIENT WEB APPLICATIONS

- **Web** evolution
  - From distributed *open hyper-text platform*...
  - ...to a Software-as-a-Service (SaaS) platform
    - distributed and open application platform, based on Web protocols
    - cloud computing
- here we focus on the **client** side
  - Rich Internet Applications
  - Web 2.0 apps, ...
  - ...



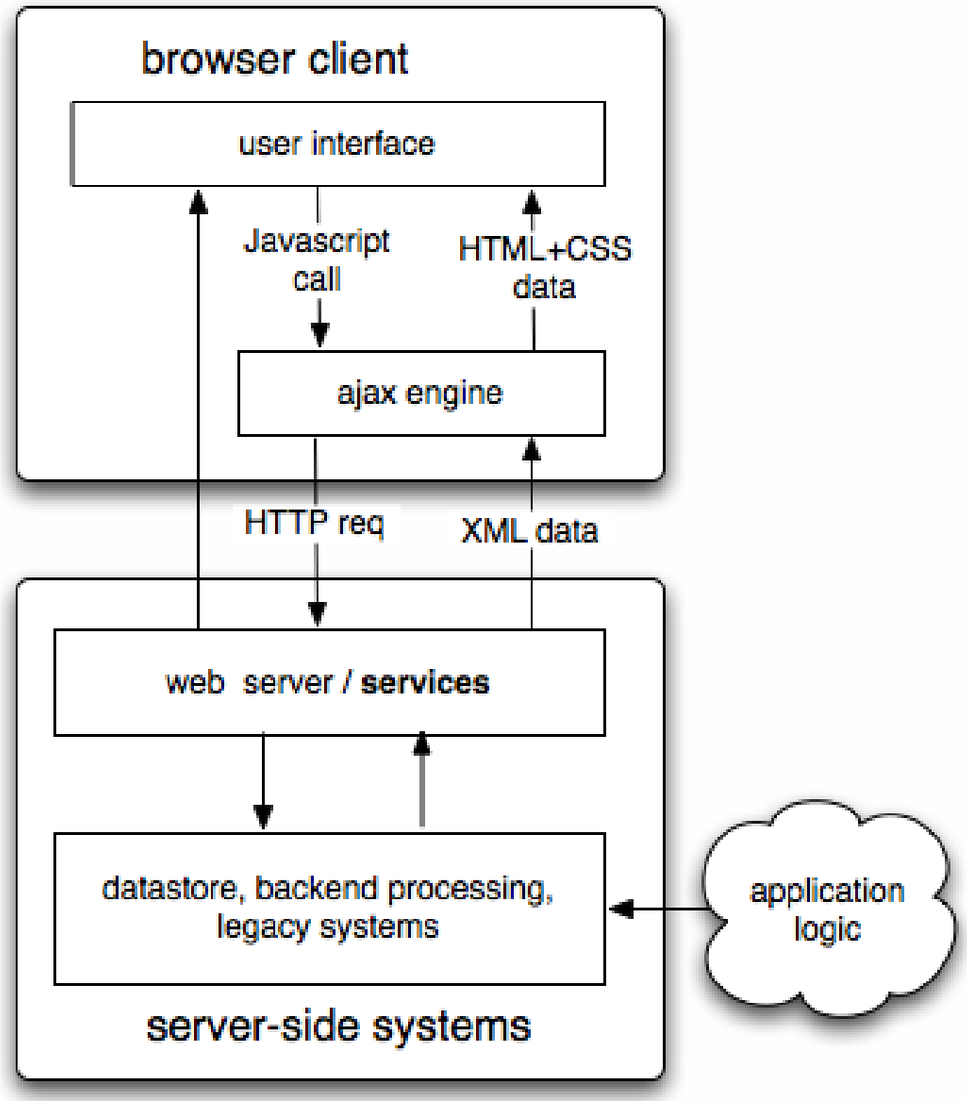
CLIENT WEB APPLICATIONS

- **Web** evolution
  - From distributed *open hyper-text platform*...
  - ...to a Software-as-a-Service (SaaS) platform
    - distributed and open application platform, based on Web protocols
    - cloud computing
- here we focus on the **client** side
  - Rich Internet Applications
  - Web 2.0 apps, ...
  - ...



CLIENT WEB APPLICATIONS

- **Web** evolution
  - From distributed *open hyper-text platform*...
  - ...to a Software-as-a-Service (SaaS) platform
    - distributed and open application platform, based on Web protocols
    - cloud computing
- here we focus on the **client** side
  - Rich Internet Applications
  - Web 2.0 apps, ...
  - ...



NEW GENERATION WEB APP: FEATURES AND COMPLEXITIES

- Features
  - application logic and responsibility distributed between the client and the server side
  - asynchronous interaction between the client parts and remote services
  - reactive user interface
  - HTML 5.0 features
    - supporting features aimed at working offline
    - supporting concurrency (WebWorkers...)
  - ...
- Increasing **complexity** in web application design and development
  - no more simple CRUD applications

MAINSTREAM MODELS AND TECHNOLOGIES: WEAKNESSES

- Scripting technology (JavaScript + AJAX ...)
  - lack of features for programming-in-the-large
    - typing, modularity, encapsulation, ...
  - callback based mechanism to deal with asynchronous events
    - *asynchronous spaghetti code*
  - low-level and error-prone support to concurrency
    - Web-worker low-level actor-based support for concurrency

- OO Frameworks (Google Web Toolkit (GWT), .NET Silverlight,...)
  - applying a full-fledge OO programming model..
    - => programming in the large features
  - ..but still no essential improvements for asynchronous events management & concurrency

APPLYING AGENT-ORIENTED PROGRAMMING

- Desiderata
  - high-level programming to deal with asynchronous events
    - user, remote services, ...
  - exploiting concurrency
    - integrating active and reactive behaviour
  - smart and flexible behaviour
    - integrating with Semantic Web
  - ...

- Good case study for agent-oriented programming
  - designing and programming web applications as multi-agent programs

EXISTING AGENT TECHNOLOGIES FOR THE WEB

- Mostly related to using agent technology on the server side
  - **Jadex Webbridge** [Pokahr and Braubach, 2008]
  - **Jack WebBot** [Agent Oriented Software Pty]
  - **JADE Gateway** agent

- Here we consider Web ***client*** application
  - the multi-agent program is running in the browser on the client side
  - ...possibly interacting with services / agents on the server side

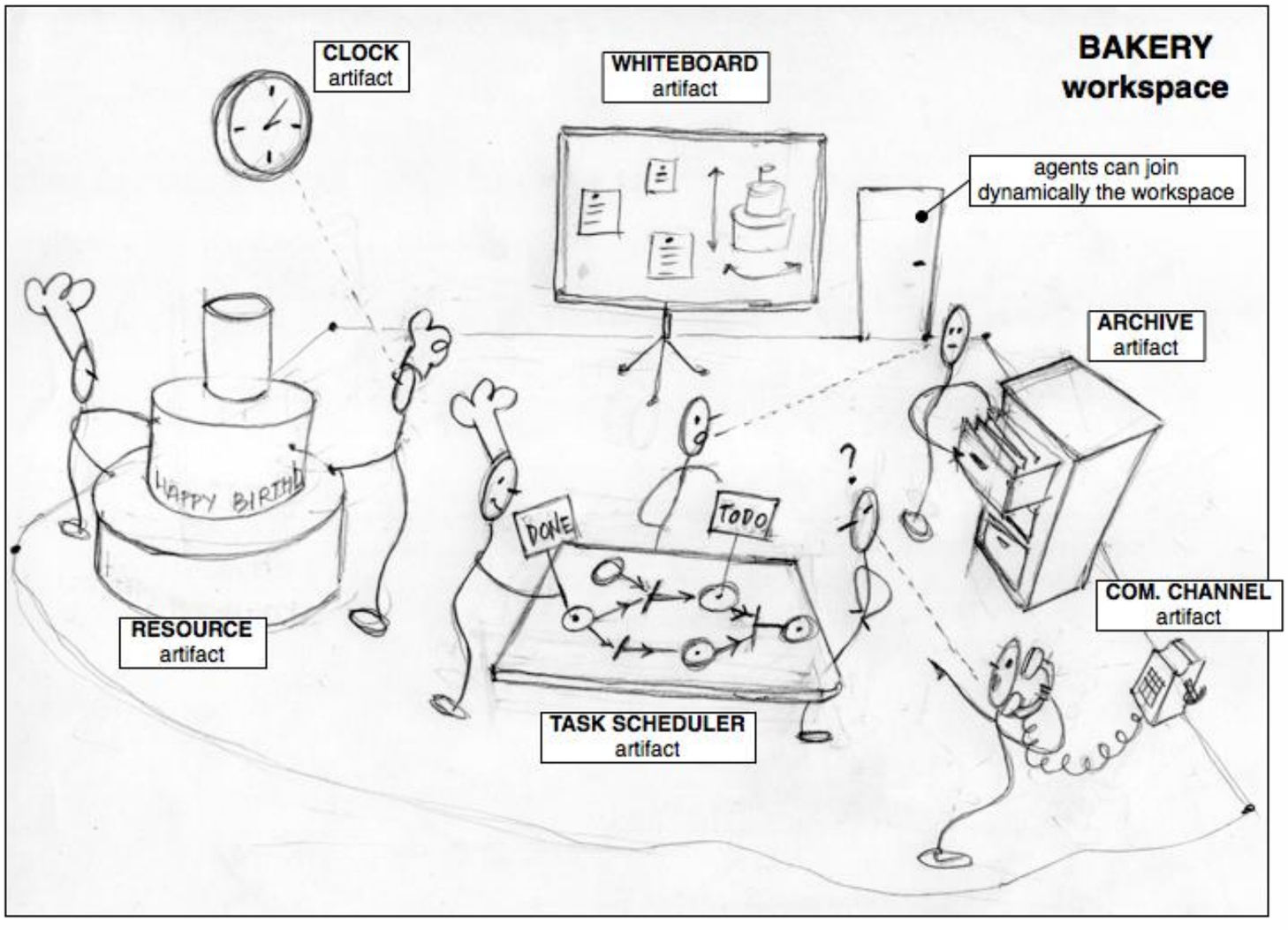
THE JaCa-WEB PROJECT

- **JaCa**
  - agent-oriented general-purpose platform for programming software systems
  - based on ***Jason*** agent programming language and **CArtAgO** Environment programming technology
    - integrating agents with a BDI architecture with environment technology

- **JaCa-WEB**
  - a framework based on JaCa to develop Client Web applications
  - exploiting (and hiding) low-level web technologies

JaCa IN BRIEF

- JaCa program
  - dynamic set of ***Jason* agents** working together an artifact-based environment
  - one or multiple **workspaces**, possibly distributed over the network
- Reference conceptual model: **A&A** (Agents & Artifacts)

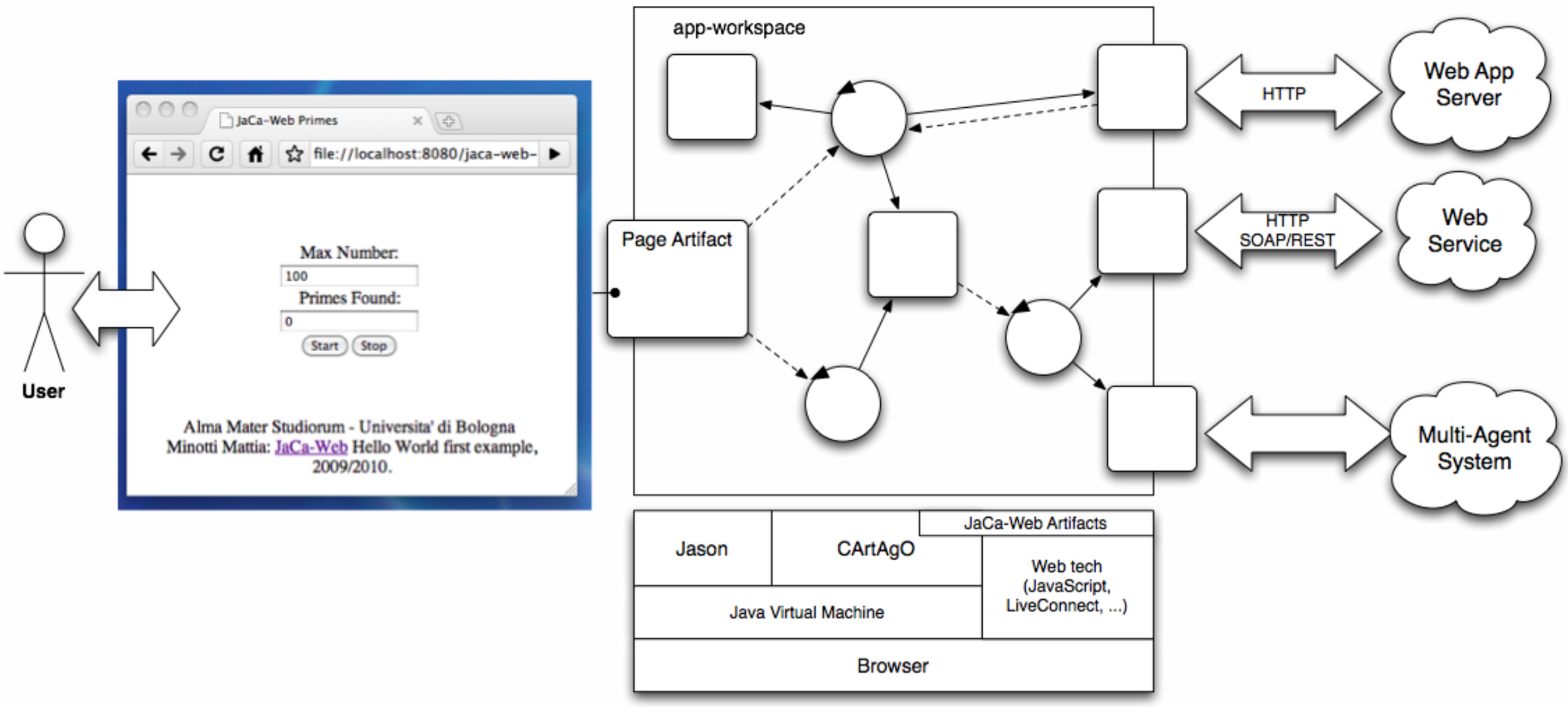


JaCa ABSTRACTIONS

- **Agents**
  - tasks/goals, plans, beliefs, actions/perception
    - BDI architecture
  - direct communication through ACL
  - indirect interaction through the environment
- Environment: **Artifacts** in **workspaces**
  - resources and tools encapsulating the functionalities that can be shared and used by agents
  - operations, observable properties and signals
- Agent/environment integration
  - action/operation mapping
  - percepts/observable properties & signal mapping

JaCa-WEB FRAMEWORK

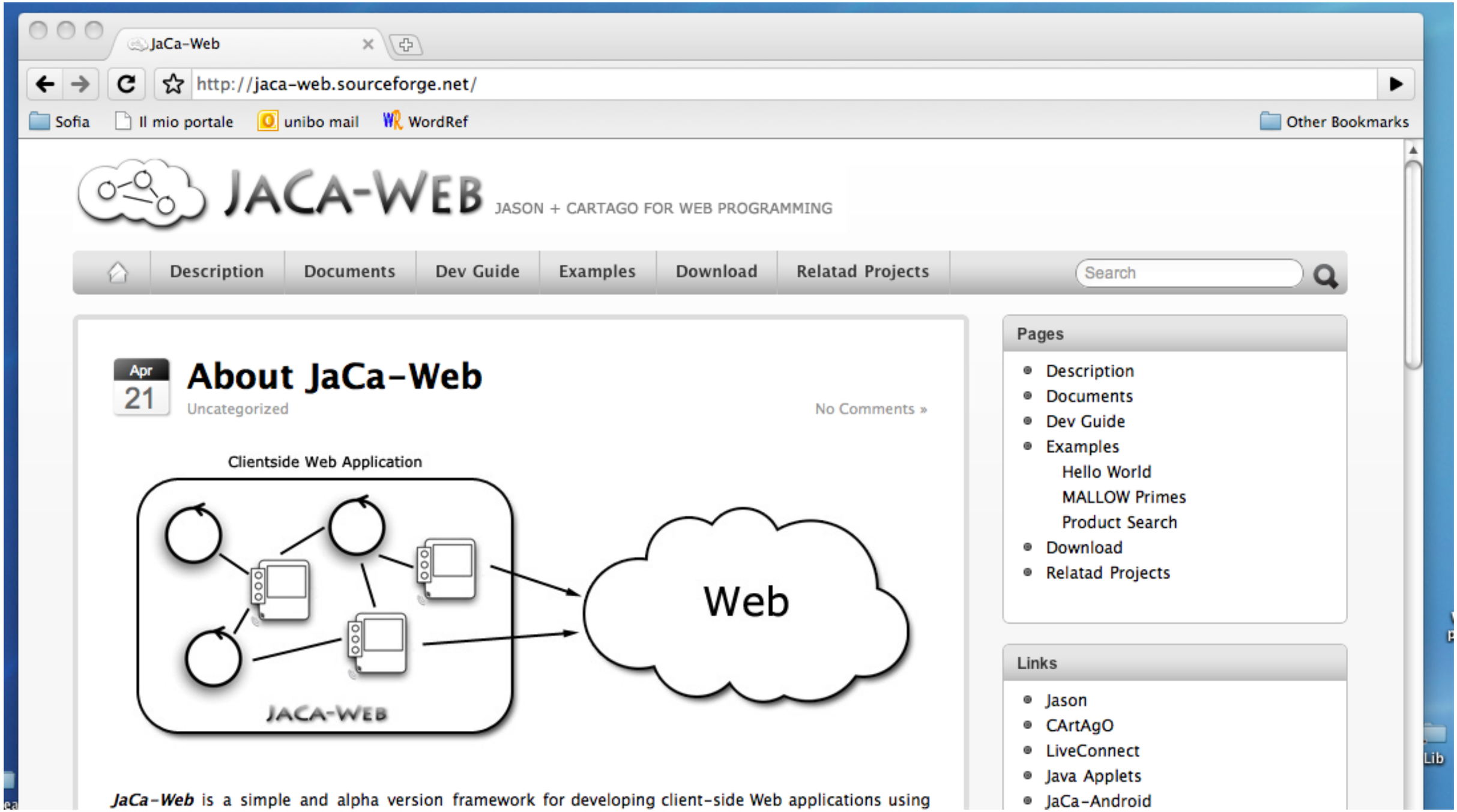
- Structuring a Client Web Client App in terms of agent-oriented abstractions



JaCa-WEB FRAMEWORK

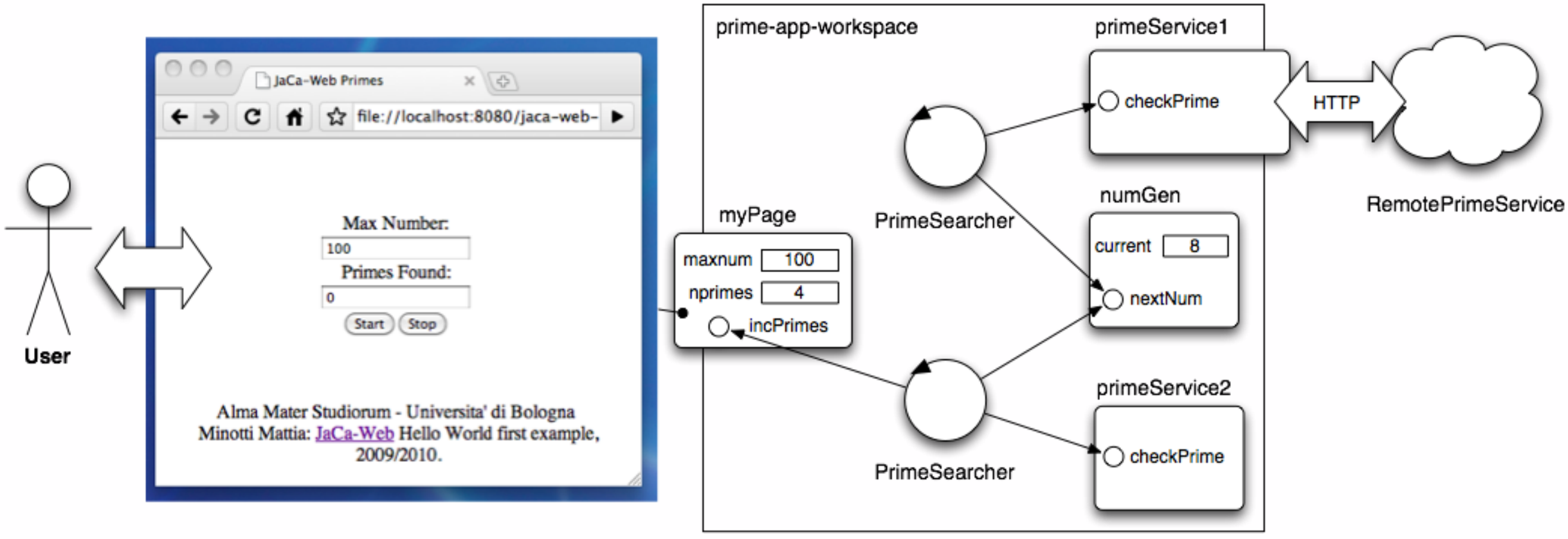
- **Agents**
  - task-oriented approach in defining Web client app behaviour
    - used to encapsulate the *control* part of the application
  - exploiting reactivity to deal with user inputs and asynchronous service interaction
- **Artifacts**
  - web and non-web resources that agents share and exploit
    - representing the *model/view* parts of the app
  - some predefined types of artifacts
    - web page & browser
    - remote services (HTTP/Web/Rest)
    - application data bases
    - ....

JaCa-Web ON THE WEB (SOURCE-FORGE)



A TOY EXAMPLE

- Concurrent interactive prime searcher app

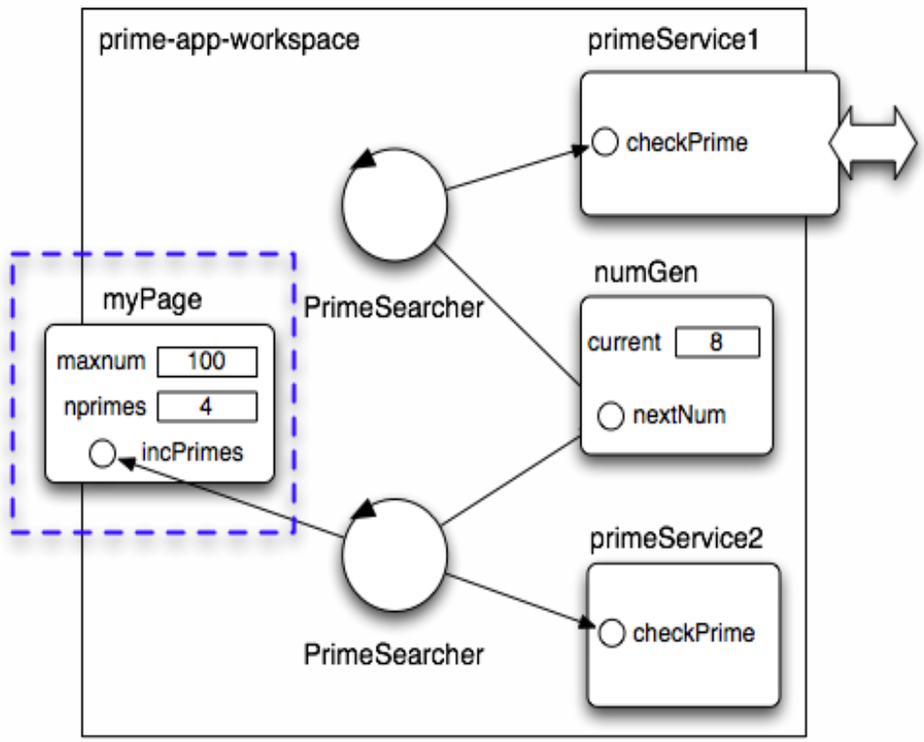


HTML

```
<html>
   <head>
      <title>JaCa-Web Primes</title>
      <script type="text/javascript" src="script/jaca-web.js"></script>
   </head>
   <body>
   <center>
                  <br/><br/><br/> Max Number:<br/>
      <input type="text" id="maxnum" value="100" />
      <br/>Primes Found:<br/>
      <input type="text" id="primes_found" value="0"/>
      <br/>
      <input type="button" value="Start" id="start" />
      <input type="button" value="Stop" id="stop" />
      <br/><br/><br/><br/>
      Alma Mater Studiorum - Universita' di Bologna <br>
      Minotti Mattia:
      <a href="http://jaca-web.sourceforge.net/">JaCa-Web</a> Prime numbers example, 2009/2010.
      </center>
      <applet            id="jacaweb" name="jacaweb" code="jacaweb.JacaWebRunner.class"
                archive="app.jar,jaca-web.jar,jason.jar,cartago.jar,c4jason.jar"
                width="0" height="0" MAYSCRIPT='true' >
         <PARAM name="mas2j_path" value="/mallow_primes/application/main.mas2j" />
         <PARAM name="page_artifact_name" value="MyPageArtifact" />
         <PARAM name="page_artifact_class" value="mallow_primes.application.MyPageArtifact" />
      </applet>
    </body>
</html>
```

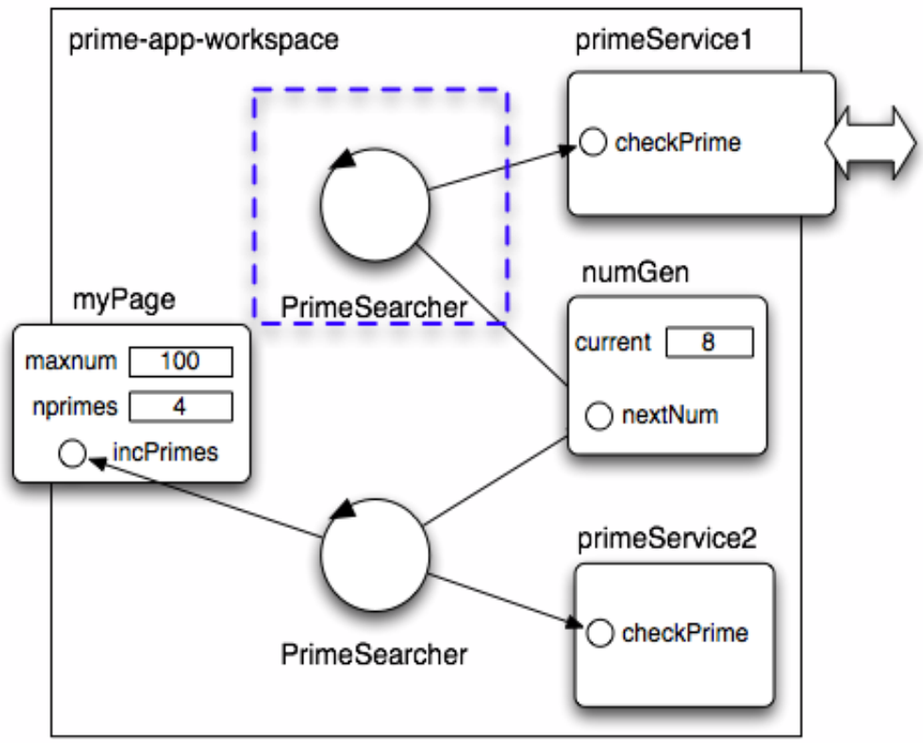
ARTIFACTS

```
public class MyPage extends PageArtifact {

  protected void setup() {
        defineObsProperty("maxnum",getMaxValue());
        linkEventToOp("start","click","startClicked");
        linkEventToOp("stop","click","stopClicked");
        linkEventToOp("maxnum","change","maxnumChange");
  }

  @OPERATION void incPrimes(){
        Elem el = getElementById("primes_found");
        el.setValue(el.intValue()+1);
  }

  @INTERNAL_OPERATION private void startClicked(){
      signal("start");
  }

  @INTERNAL_OPERATION private void stopClicked(){
      signal("stop");
  }

  @INTERNAL_OPERATION private void maxnumChange(){
        ObsProperty prop = getObsProperty("maxnum");
        obs.updateValue(getMaxValue());
  }

  private int getMaxValue(){
       return getElementById("maxnum").intValue();
  }
```



PRIME SEARCHER AGENT

```
!setup.
+!setup
  <- focusByName("MyPage");
     makeArtifact("primeService1",
                  "RemotePrimeService");
     makeArtifact("numGen","NumGen").

+start
  <- focusByName("primeService1");
     focusByName("numGen");
     !!checkPrimes.

+!checkPrimes
  <- nextNum(Num);
     !checkNum(Num).

+!checkNum(Num): maxnum(Max) & Num <= Max
  <- checkPrime(Num);
     !checkPrimes.

+!checkNum(Num)
  <- maxnum(Max) & Num > Max.

+is_prime(Num) <- incPrimes.

+stop <- .drop_intention(checkPrimes).
```



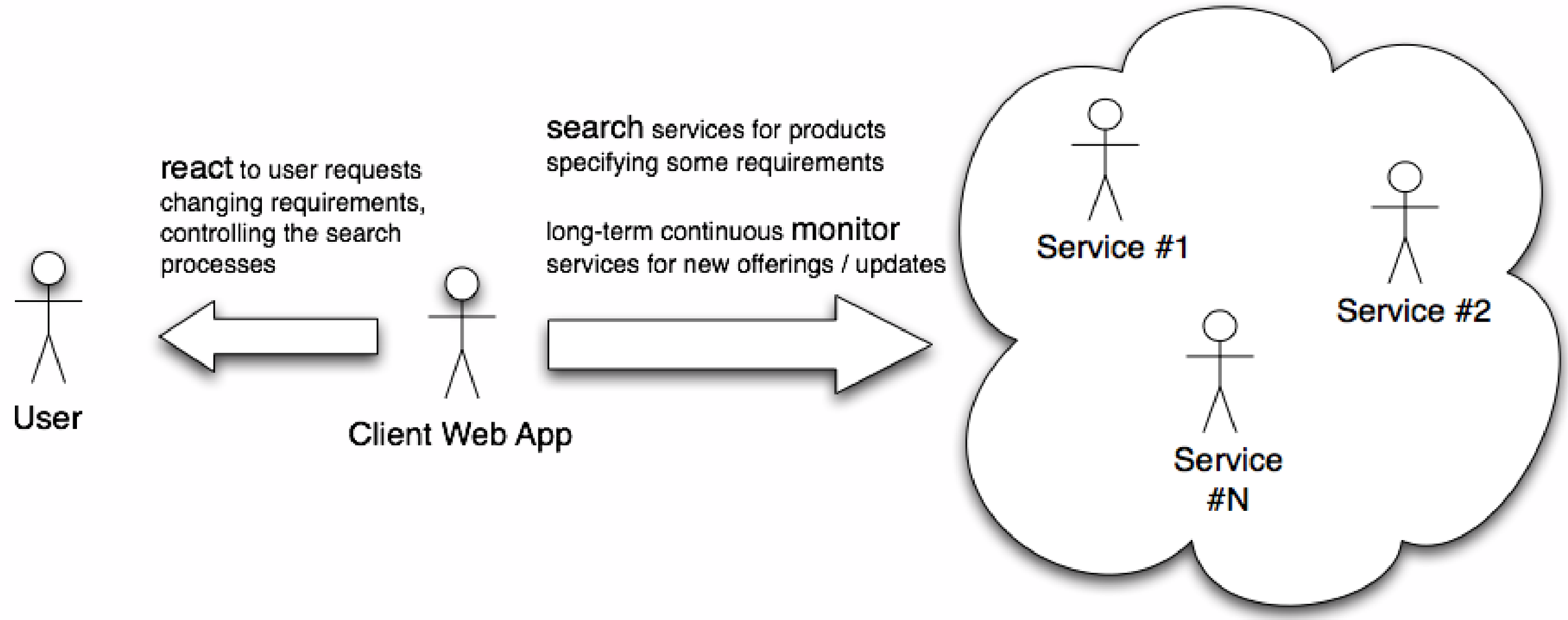
KEY POINTS

- Tackling complexity with a proper abstraction level
  - handling **concurrency** and **reactive/asynchronous** behaviour in a uniform and straightforward way
    - no need to use error-prone low-level mechanisms/idioms
    - integrating a *thread*-oriented and *event*-oriented programming style

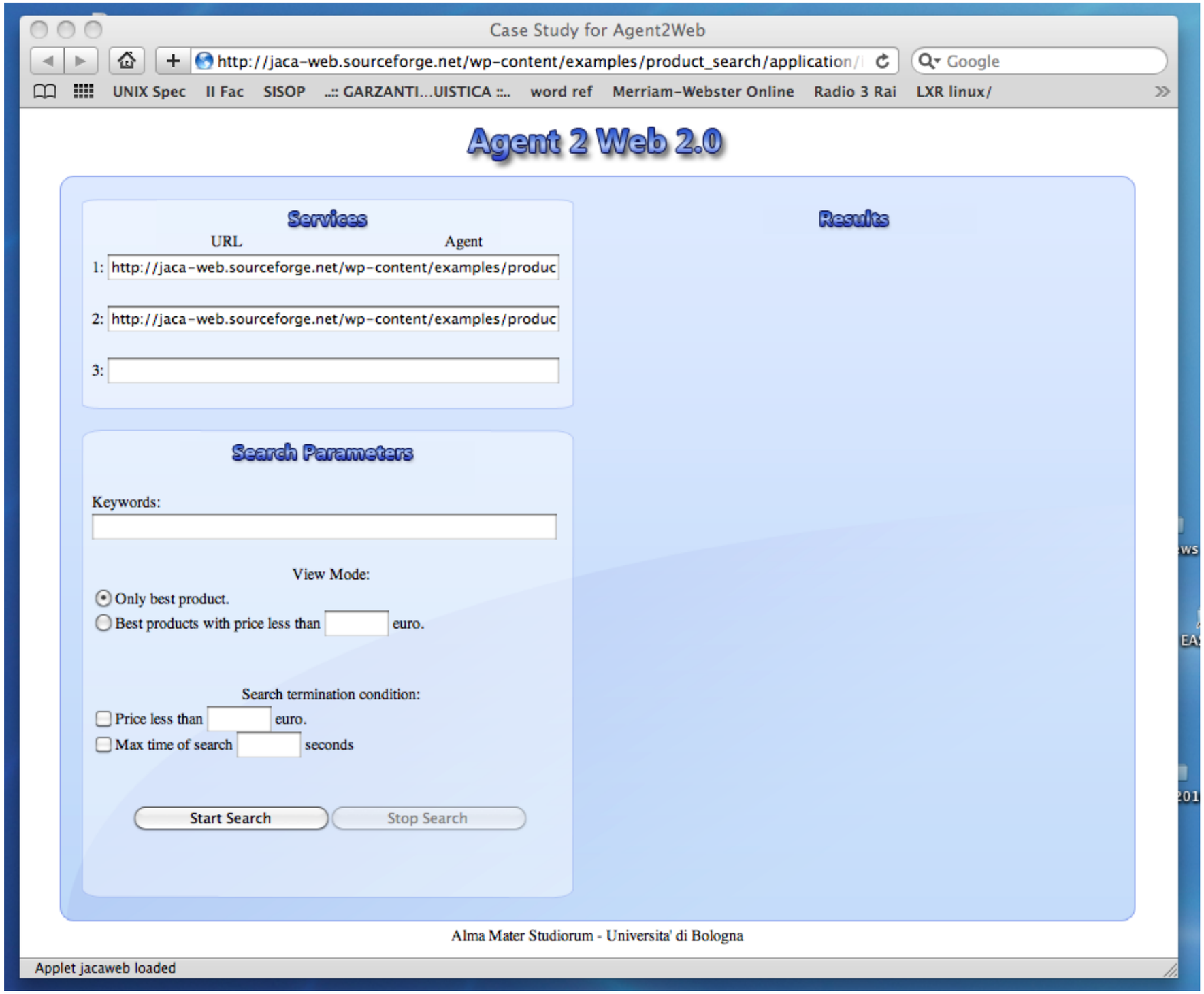
- Improving engineering
  - separation of concerns
    - task/goal-oriented and function-oriented parts
  - coarse-grain modularity and encapsulation
    - in terms of agents and artifacts
  - reducing the gap between design and implementation
    - supporting a task/goal-oriented design

CASE STUDY

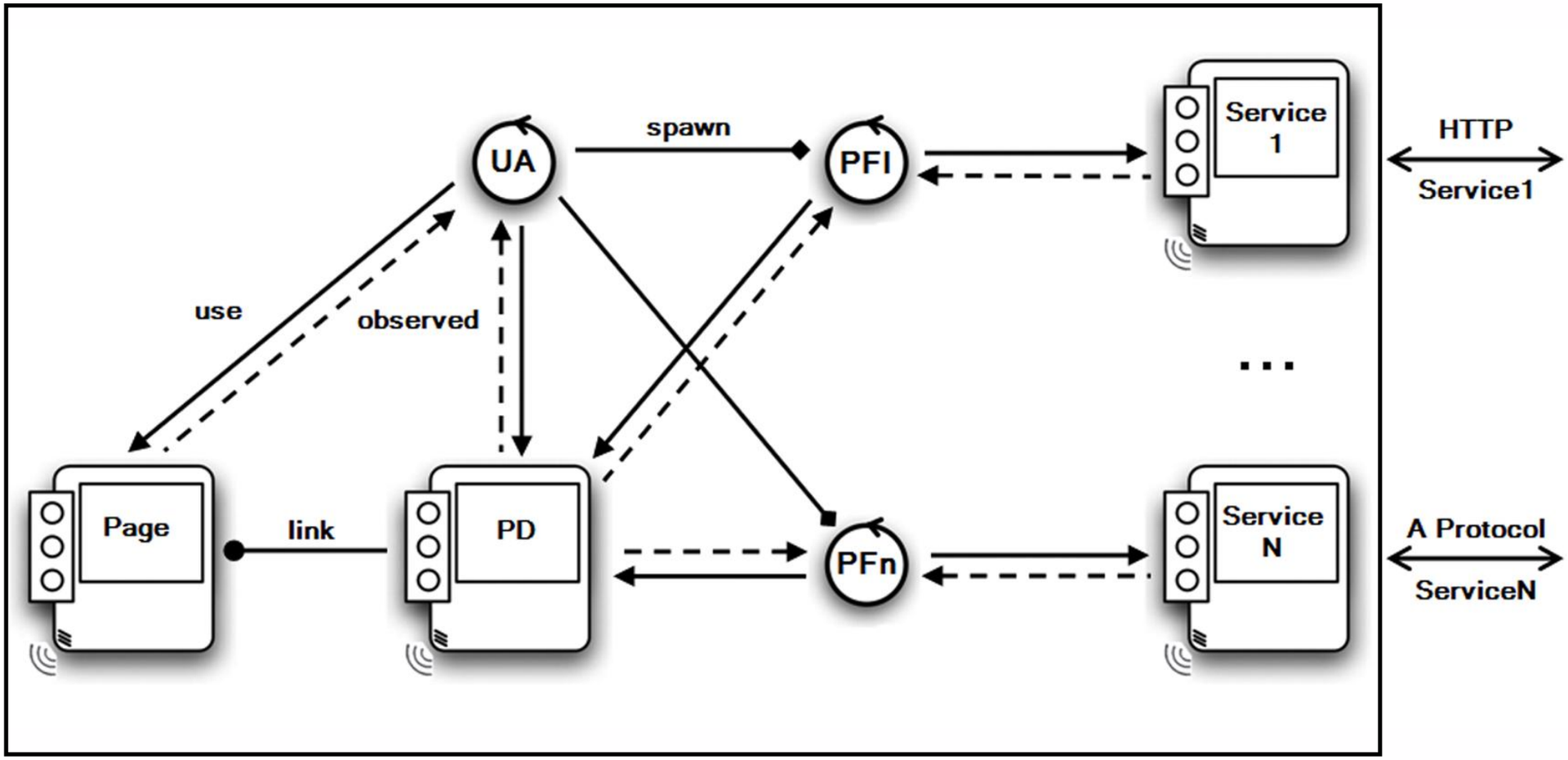
- Online search/monitoring of products to buy on the Web



(AVAILABLE ON THE WEB)



ARCHITECTURE



- Agents
  - User Assistant Agent (UA), Product Finder Agents (PF)
- Artifacts
  - Product Directory (PD), Page, Services

PRODUCT FINDER AGENT SNIPPET

```
+searchState("start")
  <- focus("service1");
     !!search.

+!search: keywords(Keywords)
  <- requestProducts(Keywords,ProductList);
     !processProducts(ProductList, ProductsToAdd, ProductsToRemove);
     addProducts(ProductsToAdd);
     removeProducts(ProductsToRemove);
     .wait({+keywords(_)},5000,_);
     !!search.

+searchState("stop")
  <-.drop_intention(search).
```

WEAKNESSES & FUTURE WORKS

- In general
  - missing a notion of type for agents and artifacts
    - detecting errors at compile time
    - exploring the notions of *inheritance*, *subtyping*, and related, in an agent oriented programming
  - improving modularity in agent definition
    - structuring the set of plans
  - improving the integration of OO data model

- Specifically to the client Web context
  - (improving the engineering of the framework)
  - integration of **semantic Web** technology
    - **JASDL [Klapiscak & Bordini, 2009]** as a starting point

ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA
SEDE DI CESENA

aliCE

WOA 2010

Developing Web *Client* Applications with JaCa-Web

Mattia Minotti, Andrea Santi, Alessandro Ricci

DEIS, Università di Bologna

Cesena (FC) Italy