SwapMe

Semantic Web Application Platform for the Mobile Ecosystem

Ora Lassila (Nokia)
Tim Berners-Lee, Daniel Jackson, David Karger, Daniel Weitzner (MIT)

January 2006
Project Goals (1)

• Build systems that offer users flexible, context- and policy-aware means to
  • access any information (local and remote)
  • manipulate (and interact with) data and environments
• Build systems that do more on behalf of users
• We target – but are not limited to – mobile devices
Project Goals (2)

• Platform for building “Semantic Web applications”
  • ubiquitous support for Semantic Web data (acquisition, transformation, storage, querying, reasoning, etc.)
  • support for context- and policy-awareness (using Semantic Web technologies)
  • MacOS “Dashboard” -like framework for small applications

• Ontology-based approach to application construction
  • functionality and services fully represented
  • automatically built workflows, goal-oriented operation
Context- and Policy-Awareness

- **Rich models of usage contexts**
  - using reasoning & rules to infer context(s)
    - current location & “task” are important dimensions of context
    - context information is used to adapt application behavior
- **Rich models of various policies**
  - privacy, access, security, etc.
  - more generally, a “policy” is a representation about how the system should behave in some future situation
    - this is a key to autonomy
  - using reasoning/rules to provide compliance/enforcement
Issues with Current Applications

• Current applications “imprison” data they “own”
  • data formats do not offer accessible semantics
  • formats are typically proprietary
  • semantics not declarative

• Ensuring interoperation introduces a high cost
  • any interaction has to be specifically designed/engineered
  • heavy emphasis on application-specific standardization

• Ad hoc interoperation is impossible
“Brave New Applications”

- Operate autonomously in “unanticipated” situations

- Exhibit robustness in the face of
  - changing, inconsistent and unexpected data
  - variations in reliability, trust

- Capable of serendipitous behavior, opportunism
“Smart Data”

- **New approach: separate data from applications**
  - data carries declarative descriptions of its semantics
  - manipulate any data with (almost) any application
    - e.g., browse photos using your calendar

- **All data available in Semantic Web formalisms**
  - shared local and distributed repositories (“triple stores”)
  - legacy data sources “exposed” as RDF (e.g., via XSLT)
  - query data via SPARQL, WilburQL, etc.
Research Questions

• What happens when data is “decoupled” from applications?
  • will traditional applications disappear?
  • will this enable ubiquitous computing?

• What is the efficient separation of concerns between the small applications, the middleware/platform and the data store(s)?

• Will “smart data” enable smarter applications?

• What is the “browser” for the Semantic Web?
  • David’s “Haystack”, Tim’s “Tabulator”, Ora’s “OINK”? 
Game Plan

• Specify compelling use cases
  • exploiting mobility, contextuality
  • e.g., supporting meetings and collaboration (with Nokia/ES)
  • e.g., personal media organization (with Nokia/M)

• Build infrastructure
  • “Dashboard”-like rapid-development framework
  • application demos

• Collaborate with other projects
  • (even others could eat our dog food)

• Capture insight about mobility and context-awareness
Existing Assets (1)

- **cwm, Wilbur, SWRP**
  - generic Semantic Web toolkits (reasoning, querying, etc.)

- **Alloy**
  - constraint solver

- **Haystack**
  - “universal” information client

- **Piggy-Bank**
  - tool for collecting and querying Semantic Web data

- **Tabulator, OINK**
  - Semantic Web “browsers”
Existing Assets (2)

- **SAMA-SE**
  - ontology-enabled dynamic user-interface generator

- **XML2RDF**
  - proxy for transforming legacy data into RDF & OWL

- **Zakim-bot, RRSAgent**
  - automated support for meetings and teleconferences

- **CALI**
  - context engine (DL, temporal reasoning and rules)

- **Rei**
  - policy engine/reasoner
Daniel Jackson
Getting Context-Aware

Attacking context
- standard approach: eliminate, by replicating context
- our approach: tolerate variation

Declarative assembly
- user indicates task: end-goal to achieve
- system suggests plan: sequence of actions to perform
Example: email

Based on discussion with Alex Ran and work by Felix Chang

Inputs

- task: ReplyToLastMsgFrom (name)
- actions: NameLookup, SortMsgsByDate, FilterMsgByAddr, Del, Reply, Send, …
- state: ...

Plan generated

- addr := NameLookup (name)
- SortMsgsByDate ()
- msgs := FilterMsgByAddr (addr)
- Reply (msgs.last)
Next steps

Challenges

• can actions & tasks be easily specified?
• is a general ontology of tasks sufficient across apps?
• can we accommodate imperfect plans?
• can planner exploit smallness of change for large states?
• can context limit search space?
Danny Weitzner
Approach to Policy-Awareness

- Departure from traditional approach to security, privacy, ownership
- Enable flexible, decentralized approach to policy management
  - local control (vs. centralized authorities)
  - rule-based permissions (vs. token-based)
- Evaluate policies with reference to:
  - user preferences
  - user data
  - web data
  - operating context
Questions? Comments?