Smart Home Planning Programs

Riccardo De Masellis, Claudio Di Ciccio, Massimo Mecella, and Fabio Patrizi

Dipartimento di Informatica e Sistemistica
Sapienza Università di Roma
Rome, Italy
www.dis.uniroma1.it/~patrizi
patrizi@dis.uniroma1.it

Sapienza Università di Roma

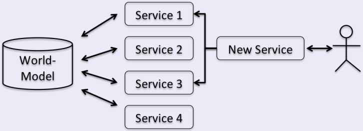
Tokyo – Japan, June 29, 2010

Service Oriented Computing and Service Composition

- SOC paradigm [ACKM04]:
  - existing software modules are fundamental blocks
  - new modules are built from existing ones
  - main advantages: software re-use; extensibility; low-cost and rapid development

Typical Scenario

- available services instructed (by clients) to execute operations
- operations affect service state
- operations affect the world-model



The SM4All Project
(Smart hoMes for All)

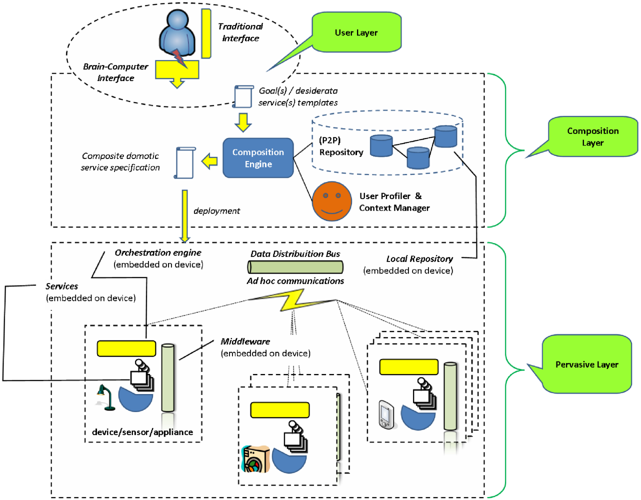
Pervasive Environments

- sensors and actuators spread throughout the house
- sensors *access* the state of the house (world-model)
- actuators *affect* the state of the house
- both have their own *internal state*

IDEA: sensors and actuators play as *services*

GOAL: assist people in carrying out desired tasks

The SM4All Architecture

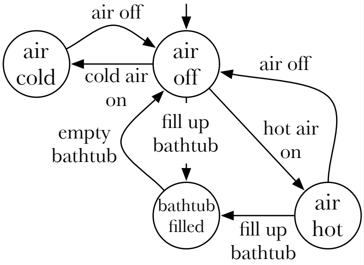


Service Abstraction

How are services described?

- *behavioral* [BCG+03] perspective: focus on dynamics (vs. in/out description)
- services offer basic operations, which affect the world-model
- operations have, in general, (partial) order constraints (protocol specification)
- captured by finite-state transition systems

Example

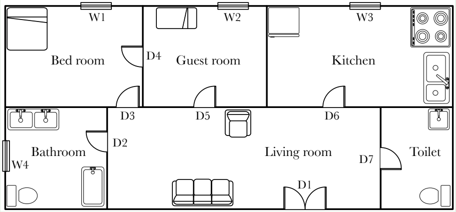


OBS: the bathtub cannot be filled if cold air is on

House Model

- A simplified model: finite-state transition system
  - finite-state (discrete and finite measures)
  - transitions triggered by service operations
- States feature propositional properties (same as in planning)

Example



All parts (e.g., window, door, etc.) controlled by some service
State: state of all components (sensors/actuators)

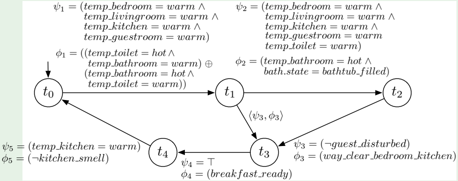
Target Service Model

Target Service: description of user's desired service

- *goal-based*: at each step (transition), the user requests a goal
- generic form: *achieve* φ *while maintaining* ψ
  - goals to achieve in the house (e.g., window open)
  - goals to maintain (e.g., keep door closed)

Essentially, a *deterministic* transition system where goals label transitions

Example



The Problem

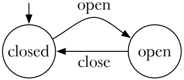
Goal-Based Service Composition Problem [DGPS10]

- Given
  - A finite set of available services
  - A finite-state *world-model*
  - A goal-based target service
- *Control* the available services so as to *realize* the target service

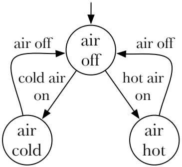
  Realizing: being always able to fulfill target goal requests

- Main novelty: request for goals, instead of operations
- Flavor of Planning:
  - transitions realized by conditional plans, through operation delegation
  - goal *routines*: requests can be chained
  - step-by-step planning fails (future requests are neglected)

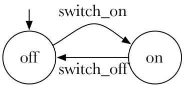
Example



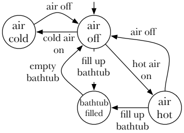
(a) door/window service



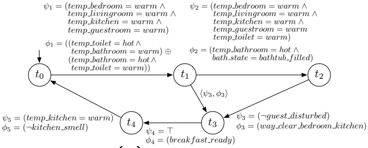
(b) house/toilet heating system



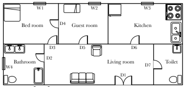
(c) kitchen ventilation



(d) bathroom tub/air controller



(e) target service



(f) world-model

- Can the available services be coordinated so as to fulfill all user goal requests?
- If so, how can it be done?

Formal Model

- The available services are usual ND transition systems $\mathcal{S} = \langle S, s_0, A, \delta \rangle$
  - S: finite set of states
  - $s_0 \in S$: initial state
  - A: set of operations
  - $\delta \subseteq S \times A \times S$: transition relation
- The world model is an ND labelled transition system $\mathcal{B} = \langle P, B, b_0, A, \rho \rangle$
  - P: finite set of propositions
  - $B \subseteq 2^P$: finite set of states
  - $b_0 \in B$: initial state
  - A: set of operations (same as above)
  - $\rho \subseteq B \times A \times B$: transition relation
- The target service is a deterministic TS with edges labelled by *propositional (goal) formulae* built from P

Observe: TSs are usually represented *compactly* (no state enumeration)

Plan-Based Simulation Relation

- How to formalize the notion of *realization*?

Definition (Plan-Based Simulation Relation)

- Consider:
  - a set of available services $\mathcal{S}_i = \langle S_i, s_{i0}, A, \delta_i \rangle$
  - a world-model $\mathcal{B} = \langle P, B, b_0, A, \rho \rangle$
  - a target service $\mathcal{T} = \langle T, t_0, G, \varrho \rangle$, with $G \subseteq PROP(P)$
- Let $\mathcal{T}_S = \mathcal{S}_1 \otimes \cdots \otimes \mathcal{S}_n$ be the asynchronous product of all available services
- $\preceq \subseteq (S_1 \times \cdots \times S_n) \times T \times B$ is a <u>*Plan-Based Simulation Relation*</u> iff:
  - $\langle \langle s_1, \ldots, s_n \rangle, t, b \rangle \in \preceq$ implies:
    - (Local realization) for all possible target transitions $t \xrightarrow{\phi} t'$ there exists a *conditional plan* (local witness!) π, *compliant* with $\mathcal{T}_S$, leading $\mathcal{B}$ to a state satisfying ϕ
    - (Preservation) for all possible $\mathcal{B}$ states b' and all possible $\mathcal{T}_S$ states $\langle s'_1, \ldots, s'_n \rangle$, both reached after the same π execution, $\langle \langle s'_1, \ldots, s'_n \rangle, t', b' \rangle \in \preceq$

This is a *coinductive* definition (gfp) –simulation– with calls to a nested *inductive* definition (lpf) –reachability)

Realizable Target Services

Intuition:

$\langle\langle s_{10}, \ldots, s_{n0}\rangle, t_0, b_0\rangle \in \preceq$: the available services can be coordinated so as to fulfill, in an *online* fashion, all sequences of goals the target service may request

Remarks:

- sequences can be *infinite*, when loops are present (naïve use of planning avoided!)
- *online*: requests not known in advance (all possible futures must be considered)

Definition (Realizable Target Service)

- A target service $\mathcal{T}$ is *realizable* by a set of available services $\mathcal{S}_1, \ldots, \mathcal{S}_n$ on a shared blackboard $\mathcal{B}$ iff a Plan-Based simulation relation $\preceq$ exists such that $\langle\langle s_{10}, \ldots, s_{n0}\rangle, t_0, b_0\rangle \in \preceq$

Plan-Based Compositions (cont.)

By choosing and executing a plan at each request, a *composition* is defined

Definition (Plan-Based Service Composition, Informal)

A *composition* is a function $comp : ((S_1 \times \cdots \times S_n) \times T \times B)^+ \times \varrho \to \Pi$ that returns a *good* plan for each target request

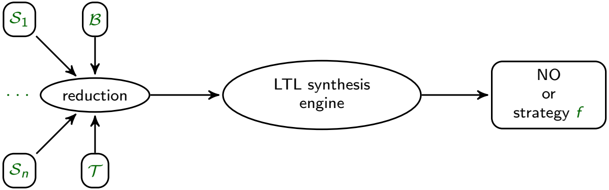
- $(S_1 \times \cdots \times S_n) \times T \times B)^+$: histories of the (whole) system
- ϱ: target requests
- Π: all conditional plans definable on $(\mathcal{S}_1 \otimes \cdots \otimes \mathcal{S}_n)$

A *good* plan is one that:

- never delegates an operation to a service unable to perform it
- guarantees the achievement of the current goal request

Computing Plan-Based Compositions

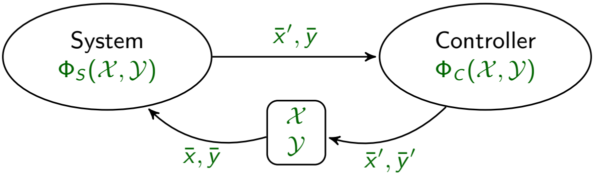
Compositions are computed by resorting to *LTL synthesis*



Advantages:

- Limit state space explosion (BDD-based approach)
- Actual technology available (e.g., TLV)

LTL Synthesis Overview



Game perspective

From current state $\langle \bar{x}, \bar{y} \rangle$ (initially, fix $\langle \bar{x}_0, \bar{y}_0 \rangle$):

- (Move) the system chooses a valuation $\bar{x}'$ for $\mathcal{X}$ propositions, that satisfies Φ_S
- (Reply) the controller chooses a valuation $\bar{y}'$ for $\mathcal{Y}$ propositions, that satisfies Φ_C
- (Play) moves and replies alternate creating (infinite) runs $\rho = \langle \bar{x}^1, \bar{y}^1 \rangle \langle \bar{x}^2, \bar{y}^2 \rangle \cdots$

Goal:

- Find a controller *winning* strategy $f : X^+ \to Y$ such that
  - all possible plays ρ *compliant* with f are s.t. $\rho \models \varphi$
  - (*compliant*: the controller plays according to f)

Intuition:

- The controller has a strategy to force φ, no matter how the system plays

LTL Synthesis Overview (cont.)

Complexity:

- For arbitrary φ, the problem is 2EXPTIME-complete [PR89]
- Generalized Reactivity (1) specifications yield an EXPTIME bound [PPS06, KPP05]

GR(1) specification form

$$\varphi = \varphi_a \wedge \bigwedge_m \Box\Diamond\phi_i \longrightarrow \varphi_r \wedge \bigwedge_\ell \Box\Diamond\psi_j$$

- φ_a: system structural assumptions (transition relation)
- φ_r: controller structural assumptions (transition relation)
- ϕ_i, ψ_j: boolean formulae

Synthesis for GR(1) formulae is enough to capture our problem!

Results

Theorem (Soundness & Completeness)

Given a composition problem instance $(\mathcal{S}_1, \ldots, \mathcal{S}_n, \mathcal{B}, \mathcal{T})$, *there exists a plan-based composition* comp *iff there exists a controller winning strategy* f *for the LTL specification* φ, *obtained from the above reduction:*

$$\varphi = (\bigwedge_n Init_{\mathcal{S}_i} \wedge Init_{\mathcal{B}} \wedge Init_{\mathcal{T}} \wedge \Box(\bigwedge_n Trans_{\mathcal{S}_i} \wedge Trans_{\mathcal{T}})) \longrightarrow (\Box good \wedge \Box\Diamond last)$$

Moreover, extracting comp from f is straightforward!

Results (cont.)

Theorem (Complexity)

The problem of checking the existence of a plan-based composition for a target service $\mathcal{T}$ by a set of available services $\mathcal{S}_1, \ldots, \mathcal{S}_n$ on a blackboard $\mathcal{B}$ is EXPTIME-complete.

Same complexity class as Conditional Planning w/ Full Observability!

Conclusions

1. A new service composition framework exploited
2. Problem reduced to LTL synthesis:
   - connection with Formal Methods: exploit MC symbolic approach
   - existing technology available
3. Sound & complete approach, same complexity class as Conditional Planning under full observability

Research Directions

1. Extensions:
   - strong fairness constraints on services/blackboard dynamics, to capture necessary eventualities (e.g., an ND operations will eventually succeed)
2. Data: what if operations have parameters and actual data are relevant to world-model and/or services?
   - this is the main open issue, as real services typically deal with data
   - often, data yield state-infiniteness: data abstraction required
   - starting point: results on services/artifacts verification&synthesis [DHPV09, PD09]
3. Efficiency (EXPTIME is hard!):
   - general viewpoint: the problem specification affects the search
   - specific viewpoint: GR(1) expressiveness more than needed
   - use/development of other tools and/or *ad-hoc* solutions
   - other, possibly incomplete, solution strategies, e.g., heuristic-based

Thank You for Listening!

Questions?

References

Gustavo Alonso, Fabio Casati, Harumi Kuno, and Vijai Machiraju.
Web Services. Concepts, Architectures and Applications.
Springer, 2004.

Daniela Berardi, Diego Calvanese, Giuseppe De Giacomo, Maurizio Lenzerini, and Massimo Mecella.
Automatic Composition of E-services That Export Their Behavior.
In *ICSOC*, pages 43–58, 2003.

Giuseppe De Giacomo, Fabio Patrizi, and Sebastian Sardiña.
Agent Programming via Planning Programs.
In *Proc. of AAMAS'10*, 2010.

Alin Deutsch, Richard Hull, Fabio Patrizi, and Victor Vianu.
Automatic Verification of Data-Centric Business Processes.
In *Proc. of ICDT'09*, 2009.

Yonit Kesten, Nir Piterman, and Amir Pnueli.
Bridging the gap between fair simulation and trace inclusion.
Information and Computation, 200(1):35 – 61, 2005.

Fabio Patrizi and Giuseppe De Giacomo.
Composition of Services that Share an Infinite-State Blackboard (Extended Abstract).
In *IJCAI'09 Workshop on Information Integration on the Web*, 2009.

Nir Piterman, Amir Pnueli, and Yaniv Sa'ar.
Synthesis of reactive(1) designs.
In *Proc. of VMCAI'06*, volume 3855 of *Lecture Notes in Computer Science (LNCS)*, pages 364–380. Springer, 2006.

Amir Pnueli and Roni Rosner.
On the Synthesis of a Reactive Module.
In *Proc. of POPL'89*, pages 179–190, 1989.